Perspectives in Memory Research

edited by Michael S. Gazzaniga

A Bradford Book
The MIT Press
Cambridge, Massachusetts
London, England

1988

Sejnowski, T. J. 1986. "Open questions about computation in cerebral cortex," in *Parallel Distributed Processing*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press, 372–389.

Sejnowski, T. J., P. K. Kienker, and G. M. Shepherd. 1985. "Simple pattern recognition models of olfactory discrimination." *Society of Neuroscience Abstracts* 11:970.

Shepherd, G. M. 1972. "The neuron doctrine: A revision of functional concepts." *Yale Journal of Biology and Medicine* 45:584–599.

Shepherd, G. M. 1974. *The Synaptic Organization of the Brain*. New York: Oxford University Press.

Shepherd, G. M. 1979. *The Synaptic Organization of the Brain*, second edition. New York: Oxford University Press.

Shepherd, G. M. 1985. "The olfactory system: The uses of neural space for a nonspatial modality," in *Contemporary Sensory Neurobiology*, M. Correia and A. A. Perachio, eds. New York: A. R. Liss, 99–114.

Shepherd, G. M. 1986. "Apical dendritic spines of cortical pyramidal cells: Remarks on their possible roles in higher brain functions, including memory," in *Synapses, Circuits, and the Beginnings of Memory*, G. Lynch, ed. Cambridge, Mass.: MIT Press, 85–98.

Shepherd, G. M., and R. K. Brayton. 1979. "Computer simulation of a dendrodendritic synaptic circuit for self- and lateral inhibition in the olfactory bulb." *Brain Research* 175:377–382.

Shepherd, G., and R. Brayton. 1987. "Logic operations are properties of computer-simulated interactions between excitable dendritic spines." *Neuroscience* 21:151–165.

Shepherd, G. M., R. K. Brayton, J. F. Miller, I. Segev, J. Rinzel, and W. Rall. 1985. "Signal enhancement in distal cortical dendrites by means of interactions between active dendritic spines." *Proceedings of the National Academy of Science USA* 82:2192–2195.

Smith, L. M., F. F. Ebner, and M. Colonnier. 1980. "The thalamo-cortical projection in *Pseudemys* turtles: A quantitative electron microscopic study." *Journal of Comparative Neurology* 190:445–461.

Spencer, W. A., and E. R. Kandel. 1961. "Electrophysiology of hippocampal neurons IV. Fast prepotentials." *Journal of Neurophysiology* 24:272–285.

Stafstrom, C. E., P. C. Schwindt, J. A. Flatman, and W. E. Crill. 1984. "Properties of subthreshold response and action potential recorded in layer V neurons from cat sensorimotor cortex in vitro." *Journal of Neurophysiology* 52:244–263.

Stone, J. 1983. *Parallel Processing in the Visual System*. New York: Plenum.

Swanson, L. W. 1981. "A direct projection from Ammon's horn to prefrontal cortex in the rat." *Brain Research* 217:150–154.

Szentagothai, J. 1975. "The module-concept in cerebral cortex architecture." *Brain Research* 95:475–496.

Traub, R. D., and R. Llinas. 1979. "Hippocampal pyramidal cells: Significance of dendritic ionic conductances for neuronal function and epileptogenesis." *Journal of Neurophysiology* 42:476–496.

Walaas, I. 1983. "The hippocampus," in *Chemical Neuroanatomy*, P. C. Emson, ed. New York: Raven, 337–358.

In: Perspectives in Memory Research
M. S. Gazzaniga, Ed., Cambridge: MIT Press, 1988

4

Learning and Representation in Connectionist Models

Terrence J. Sejnowski and
Charles R. Rosenberg

Expert performance is characterized by speed and effortlessness, but this fluency requires long hours of effortful practice. We are all experts at reading and communicating with language. We forget how long it took to acquire these skills because we are now so good at them and we continue to practice every day. As performance on a difficult task becomes more automatic, it also becomes more inaccessible to conscious scrutiny. The acquisition of skilled performance by practice is more difficult to study and is not as well understood as memory for specific facts (Anderson 1982; Norman 1982; Squire 1986, Tulving 1985).

In connectionist models information is represented as patterns of activity in a large number of simple processing units. Memory and processing are closely intertwined in a network. Information can be stored by changing the connection strengths or weights on the links between the processing units. By studying the properties of relatively simple connectionist models, researchers may be able to gain insights into the different ways information processing is organized in the nervous system.

The earliest network models of associative memory were based on correlations between input and output patterns of activity in linear processing units (Hinton and Anderson 1981). These models have several features that make them attractive: The synaptic strengths are computed from information available locally at each synapse in a single trial; the information is distributed in a large number of connection strengths; the recall of stored information is associative; and the network can generalize to new input patterns that are similar to stored patterns. There are also severe limitations with this class of linear associative matrix models, including interference between stored items, especially between ones that are related, and inability to make decisions that are contingent on several inputs. New network models and network learning algorithms have been introduced recently that overcome some of the shortcomings of the associative matrix models of memory. These learning algorithms require many training examples to create the internal representations needed to perform a task skillfully and to generalize properly, which makes this type of learning a candidate model for skill acquisition.

4.1 *Neural Network Models of Learning*

Organizing Principles

Neural Networks Neurons are highly specialized processing units. Their relatively slow processing time is compensated for by their large number and high connectivity. There are many types of neurons that have highly specific patterns of connectivity. Some are primarily inhibitory; others are primarily excitatory. Unfortunately the detailed patterns of connectivity in the cerebral cortex have not yet been determined. Processing within neurons can be complex, although within the basic limitations on speed and accuracy imposed by the biophysical properties of ions and membranes. The dendrites in some neurons integrate incoming information through nonlinear spatiotemporal interactions between synapses. Synaptic strengths are variable on many time scales and can facilitate or habituate with activity. The anatomical arrangements found in the cerebral cortex are outlined by Crick and Asanuma (1986), and the physiological properties of cortical cells are summarized by Sejnowski (1986).

The degree of neural detail that should be included in a model depends on the level under investigation. Biophysical properties may be crucial when modeling synaptic plasticity, but only a general rule for modification may be needed to model information storage at the circuit level. The style of processing and memory, such as the degree to which information is localized or distributed in the network, could well be general properties, whereas the actual codes used are probably specific to the detailed circuits. If there are no general properties of cortical processing, then nothing short of detailed simulations of actual circuits will yield any insights, but there is hope that at least some general insights will be possible. Churchland (1986) has emphasized the importance of computational network models in providing generalizations and guidance at both the neural and cognitive levels of description.

As a first step toward understanding neural networks, we study network models constructed from simple processing units that have only the most basic properties of neurons and attempt to explore their computational capabilities: What are the possible ways to represent

sensory information in a collection of these units? What are the computational capabilities of different patterns of connectivity in the network? What computations can the network not perform? Even the simplest networks have complex behaviors that are not easy to describe analytically, so much of the research is empirical and exploratory. Also, there are so many architectures—the number of layers, feedback between layers, and local patterns of connectivity—that much guidance is needed from the general organization of cortical circuits, such as the columnar organization of the cerebral cortex and the hierarchical arrangements of cortical mappings (Adrian 1953; Hubel and Wiesel 1962; Mountcastle 1978; Allman et al. 1983; Van Essen and Maunsell 1983). Once we have gained some insight into the capabilities of these simple models, we can compare their performance with human performance on similar tasks and continue to improve the models.

Representations In this chapter we present a network model that pronounces English text by transforming letters into elementary speech sounds, or phonemes. How many neurons are involved in the representation of letters and phonemes in the cerebral cortex? A related question is, How much overlap is there between the populations of neurons? To be more specific, when the word "cat" is pronounced, how localized is the representation for the production of the sound of the letter "a" and how different is it from the sound of the letter "a" in "gate"? Almost nothing is known about these issues in part because recordings have not been made from cortical neurons in humans. Two extreme possibilities are that each item is assigned to a single neuron, the so-called grandmother cell hypothesis (Barlow 1972; Feldman 1986), or that a large number of the neurons in a brain area are used to represent an item, sometimes called the holographic hypothesis (Longuet-Higgins 1968; Willshaw 1981). Almost certainly the number of neurons involved is intermediate between these extremes and depends on the item.

The nature of internal representations in the transformations of letters to sounds can be studied by constructing network models that can perform the same task. The networks we present here are much too simple to serve as a literal model for the real neural networks in the human speech areas. However, we have been able to explore many interesting questions, including the properties of distributed internal representations and their consequences for learning strategies. Several general principles emerge based on the qualitative similarities between the performance of the network model and human abilities.

Associative Matrix Models

The goal of early models of memory (Steinbuch 1961; Anderson 1970; Kohonen 1970; Longuet-Higgins 1968) was to perform content-addressable recall of information represented as vectors. Given an input vector ι_b and an associated output vector o_a, the correlation matrix is defined as

$$K_{ab} = \varepsilon o_a \iota_b, \tag{4.1}$$

where ε is the strength of the association. If ι_b is identified with the rate of firing of the bth presynaptic element and o_a is identified with the rate of firing of the ath postsynaptic element, then K_{ab} can be computed after modifying the synapses between the input and output neurons according to the learning rule suggested by Hebb (1949), which states that the strength of the synapse should increase whenever there is a simultaneous presynaptic spike and a postsynaptic spike. An important property of the correlation matrix is that it depends only on information that is available locally at a synapse. Nonlocal modification rules that require information from disparate parts of a network are more difficult to implement.

Each component of the input and output vectors is identified with the firing rate of a neuron. The associative matrix model assumes that the output firing rate is a linear summation of all the weighted inputs. Given an input vector η_b, the output vector ϕ_a is given by

$$\phi_a = \varepsilon \sum_{b=1}^{n} K_{ab} \eta_b, \tag{4.2}$$

where n is the number of components of the input vector. By substituting the expression for K_{ab} in equation (4.2), we can rewrite the output vector as

$$\phi_a = \varepsilon o_a \sum_{b=1}^{n} \iota_b \eta_b. \tag{4.3}$$

Thus the output of the network is proportional to the stored output vector and the amplitude depends on the inner product or overlap between the input vector and the stored input vector.

Several pairs of inputs and outputs can be stored this way in the same network:

$$K_{ab} = \sum_{\alpha=1}^{A} \varepsilon_\alpha o_a^\alpha \iota_b^\alpha, \tag{4.4}$$

where o_a^α and ι_b^α are A pairs of input and output vectors, respectively, and ε_α is the association strength of the αth pair. The output vector

can similarly be related to the stored vectors:

$$\phi_a = \sum_{\alpha=1}^{A} \varepsilon_\alpha o_a^\alpha \sum_{b=1}^{n} \iota_b^\alpha \eta_b. \tag{4.5}$$

However, as the number of stored vectors increases, so does the interference between them (Anderson 1970). Crosstalk between the stored input vectors can be minimized by orthogonalizing them (Kohonen 1984).

Hebbian synaptic plasticity is probably the simplest local rule that can be used for associative storage and recall of information. Evidence supporting Hebbian plasticity has recently been found in the hippocampus (Kelso et al. 1986), and detailed correlation matrix models of the hippocampus are now being explored (Lynch 1986; Rolls 1986; McNaughton and Morris 1987). However, there are many other uses for Hebbian synaptic plasticity, such as plasticity during development (Linsker 1986), unsupervised learning (Sutton and Barto 1981; Tesauro 1986; Finkel and Edelman 1985), and rapid changes in the topology of a network (von der Malsburg and Bienenstock 1987). As a consequence, experimental evidence for Hebbian modification of synaptic strength does not necessarily imply associative storage.

Numerous variations have been proposed on the conditions for Hebbian plasticity (Levy et al. 1984). One problem with any synaptic modification rule that can only increase the strength of a synapse is the eventual saturation of the synaptic strength at its maximum value. Nonspecific decay is one solution to this problem. Sejnowski (1977a,b) suggested that specific decreases in the strength of a plastic plastic synapse should be considered and proposed that the change in strength of a plastic synapse should be proportional to the covariance between the presynaptic firing and the postsynaptic firing:

$$K_{ab} = \sum_{\alpha=1}^{A} \varepsilon_\alpha (o_a^\alpha - \bar{o}_a)(\iota_b^\alpha - \bar{\iota}_b), \tag{4.6}$$

where $\bar{o}_a$ is the average firing rate of the output neuron and $\bar{\iota}_b$ is the average firing rate of the input neuron [see also Chauvet (1986)]. According to this modification rule, the strength of the synapse should increase if the firings of the presynaptic and postsynaptic elements are positively correlated, decrease if they are negatively correlated, and remain unchanged if they are uncorrelated. Evidence for a decrease in the strength of synapses in the hippocampus under the predicted conditions has recently been reported by Levy et al. (1983). Similar modification rules have also been suggested for plasticity during development (Cooper et al. 1979; Bienenstock et al. 1982).

Improvements have recently been made to associative matrix models by introducing feedback connections, so that they are autoassociative, and by making them nonlinear (Anderson and Mozer 1981; Sejnowski 1981; Hopfield 1982; Kohonen 1984; Toulouse et al. 1986). However, this class of models still has a severe computational limitation in that all the processing units in the network are constrained by either the inputs or the outputs, so that there are no free units that could be used to form new internal representations. What representations should be used if the network is deeply buried in the association cortex far from sensory inputs and motor outputs? Some other principles must be specified for forming these internal representations. Nevertheless, given that good representations already exist, the associative matrix model is still a viable one for the fast storage of novel events and items.

Nonlinear Processing Units

In the model neuron introduced by McCulloch and Pitts (1943), the output could only take the value 0 or 1, like the all-or-none nature of the action potential. This binary model does not take into account the graded responses of neurons, which can be expressed as an average rate of firing. There are two ways to make the output of the processing unit graded. First, the output of the processing unit can be made probabilistic, with a probability proportional to its average rate of firing. Second, the output of a processing unit can be made a real number between 0 and 1. Both of these possibilities are illustrated in this section.

The output function of a more realistic model neuron is shown in figure 4.1. This function has a sigmoid shape: It monotonically increases with input; it is 0 if the input is negative; and it asymptotically approaches 1 as the input becomes large. This roughly describes the firing rate of a neuron as a function of its integrated input: If the input is below threshold, there is no output, the firing rate increases with the input, and it saturates at a maximum firing rate. The behavior of the network does not depend critically on the details of the sigmoid function, but the one we used is given by

$$s_i = P(E_i) = \frac{1}{1 + e^{-E_i}}, \tag{4.7}$$

where s_i is the output of the ith unit and the total input E_i is

$$E_i = \sum_j w_{ij} s_j, \tag{4.8}$$

where w_{ij} is the weight from the jth to the ith unit. The weights can

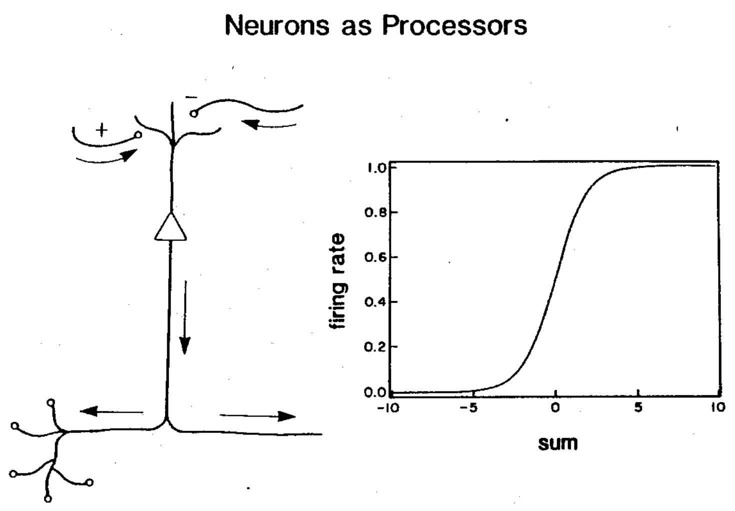

Figure 4.1
(Left) Schematic model of a processing unit receiving inputs from other processing units. (Right) Transformation between summed inputs and output of a processing unit as given by equation (4.7).

have positive or negative real values, representing an excitatory or inhibitory influence.

In addition to the weights connecting them, each unit also has a threshold. In some learning algorithms the thresholds can also vary. To make the notation uniform, we implemented the threshold as an ordinary weight from a special unit, called the true unit, that always has an output value of 1. This fixed bias acts like a threshold whose value is the negative of the weight.

The properties of the nonlinear processing units used here have some properties that make them similar to real neurons: (1) the integration of diverse excitatory and inhibitory signals arriving from other units, although with low accuracy; (2) an output signal that is a nonlinear transformation of the total integrated input, including a threshold; and (3) a complex pattern of interconnectivity. Many other properties of neurons are not taken into account but could be incorporated into subsequent models. The goal here is to explore the processing capabilities of the simplest classes of nonlinear networks, particularly those properties that arise through the patterns of connections in the network.

Nonlinear Networks with One Layer of Connections

In a network of processing units a subset receives information from outside the network while another subset provides the output from the network. Patterns of activity in the group of input units are transformed into patterns of activity in the output units by direct connections and through connections with additional internal units that play the role of interneurons. In general, it is difficult to analyze the performance and computational capabilities of nonlinear network models, but by making restrictions on the connectivity, it is possible to make progress. The small networks that we can study at present should be considered part of a larger system.

When there are feedback connections in a network, the units may reverberate without settling down to a stable output. In some cases oscillations may be desirable, but otherwise special provisions must be made to suppress them. One method that has been thoroughly explored is the use of symmetric connectivity. Networks with reciprocal symmetric connections, first introduced by Hopfield (1982) in the context of binary-valued processing units, were the starting point for the study of learning algorithms in Boltzmann machines by Hinton and Sejnowski (1983). Another method, extensively studied by Grossberg (1976), is the use of lateral shunting inhibition. But it is easiest to avoid oscillations by not considering any feedback connections.

In a feed-forward network there is no dynamic feedback so that information can flow only from the input layer to the output layer. The simplest class of feed-forward networks are ones that have no internal or "hidden" units. In this case each output unit acts independently in response to input patterns in its "receptive field," defined here as the group of input units that drives the output unit, in analogy with the concept of a receptive field for sensory neurons. The output unit is most strongly driven by patterns of activity in its receptive field that are congruent to the excitatory connections and that avoid the inhibitory ones.

A simple learning procedure exists for automatically determining the weights in a single-layer feed-forward network. It is an incremental learning procedure that requires a teacher to provide the network with examples of typical input patterns and the correct outputs; with each example the weights in the network are slightly altered to improve the performance of the network. If a set of weights exists that can solve the classification problem, then convergence theorems guarantee that such a set of weights will be found.

These learning procedures are error correcting in the sense that only information about the discrepancy between the desired outputs provided by the teacher and the actual output given by the network is used to update the weights. The LMS algorithm of Widrow and Hoff (1960) applies to units that have continuous-valued outputs, and the

perceptron learning algorithm of Rosenblatt (1959) applies to binary-valued units. The LMS algorithm is described here. Define the difference between the desired outputs s_i^* and the actual outputs s_i as

$$\delta_i = s_i^* - s_i. \tag{4.9}$$

The LMS learning algorithm requires that the weight from input unit s_j to the ith output unit should be altered by

$$\Delta w_{ij} = \delta_i s_j. \tag{4.10}$$

This is a gradient descent procedure because on each step the squared error averaged over all input patterns is reduced.

There is an interesting relationship between this error-correcting procedure and the Rescorla-Wagner theory for classical conditioning. Rescorla and Wagner (1972) state that "organisms only learn when events violate their expectations. Certain expectations are built up about the events following a stimulus complex; expectations initiated by the complex and its component stimuli are then only modified when consequent events disagree with the composite expectation" (p. 75). Thus it is the difference between the expected and the actual outcomes that determines whether strengths are modified. Sutton and Barto (1981) have shown that the mathematical formalism introduced by Rescorla and Wagner is identical with the Widrow-Hoff LMS algorithm.

Recently Gluck and Bower (1986, 1987) have applied the LMS algorithm to category learning in humans. In three experiments subjects learned to categorize diseases in hypothetical patients from patterns of symptoms. The adaptive network model was a better predictor of human performance than probability matching, exemplar retrieval, or simple prototype matching. The model correctly predicted a counterintuitive phenomenon called base-rate neglect that has been frequently observed in studies of likelihood judgments: When one disease is far more likely than another, the model predicts that subjects will overestimate the diagnostic value of the more valid symptom for the rare disease. Thus the subjects consistently overestimated the degree to which evidence that was representative or typical of a rare event was actually predictive of it (Kahneman and Tversky 1972).

The patterns that can be correctly classified with a one-layer network are limited to those that are geometrically equivalent to regions of a vector space bounded by a plane (Minsky and Papert 1969). Single-layer networks are severely limited in the difficulty of the problem that they can solve, but this deficiency can be partially overcome by preprocessing the inputs through a layer of units that serve as feature detectors so that the information needed to solve the problem is made explicitly available (Rosenblatt 1959; Gamba et al. 1961). The required features may be different for each problem.

An impressive example of how clever coding can turn a difficult problem into one that can be learned in one layer of weights is the study of verb learning by Rumelhart and McClelland (1986a). The goal of their network was to take as input English verbs and produce as output their past tenses. Their coding scheme decomposed the ordered string of letters in words into unordered triples. These triples in turn were coded into patterns on 460 input units. The same coding was used for the 460 output units, each of which received a connection from all the input units, making a total of 231,600 weights. Their network was equivalent to a single-layer Boltzmann machine, which we discuss in the next section.

One problem with single-layer networks is the lack of internal degrees of freedom. Can the learning algorithm be generalized to networks with more than one layer of weights? If so, then the need to hand-code the features for each problem would be alleviated and much more difficult problems could be solved by the same type of supervised learning paradigm. It had been thought for many years that such a learning algorithm was not possible for multilayered networks [Minsky and Papert (1969, p. 232); see also Arbib (1987)].

Nonlinear Network Models with Hidden Units

A network without hidden units is limited in what it can learn. Adding a single intermediate layer of hidden units suffices to perform any desired transformation. Consider, for example, the case of binary units. If there are N input units, then there are 2^N possible input patterns. Dedicate one hidden unit to each of these input patterns and connect it to the input units in the following way: Set the weight from an input unit to the hidden unit to $+1$ if the input unit is on or to -1 if the input unit is off, and set the threshold of the hidden unit to the total number of input units that are on. For any given input pattern only one hidden unit will be activated, which in turn can activate any desired output pattern. In this way any problem can be solved, but at the expense of a huge number of hidden units that grows exponentially with the number of input units. With continuous-valued units the analysis is more difficult, but similar theorems can be proved (Kolmogorov 1957; Palm 1978, 1979).

In practice, only a small subset of all possible transformations are ever needed and only a small number of hidden units are available. The challenge is to find the appropriate set of hidden units for each problem. One possibility is to have the network discover the proper

features without supervision from a teacher. There are several unsupervised learning procedures that can automatically model structure from the environment (Kohonen 1984; Grossberg 1976; Rumelhart and Zipser 1985; Pearlmutter and Hinton 1986). One problem with unsupervised learning is that all the hidden units may discover the same features. Competition through mutual inhibition is one solution that enforces diversity (Feldman 1982), and others have been suggested (Reggia 1985; Baum et al. 1987). Another problem is that not all the structure in the inputs may be relevant to the solution of a particular problem. Feedback of information from the environment about the desired performance is needed.

One class of supervised learning algorithms for multilayered networks uses reinforcement signals from a teacher that tell the network whether or not the output is correct (Sutton and Barto 1981; Barto 1985; Klopf 1986; Tesauro 1986; Gluck and Thompson 1986). This is the minimum amount of information needed to help direct the hidden units toward good features, but there is so little information that the networks improve slowly and hesitatingly. Recently a new class of algorithms was discovered that directly generalizes the class of error-correcting learning procedures to multilayered networks. Two examples are reviewed here: the Boltzmann machine and backpropagation. [See also Arbib (1987) for a review that includes a valuable historical perspective on earlier work.]

Boltzmann Machines Hinton and Sejnowski (1983, 1986) introduced a stochastic network architecture, called the Boltzmann machine, for solving optimization problems (Marr and Poggio 1976; Ballard et al. 1983; Hopfield and Tank 1986). The processing units in a Boltzmann machine are binary, like the perceptron, but they are updated probabilistically using the same output function in figure 4.1. As a consequence, the internal state of a Boltzmann machine fluctuates even for a constant input pattern. The amount of fluctuation is controlled by a parameter that is analogous to the temperature of a thermodynamic system. Fluctuations allow the system to escape from local traps into which it would get stuck if there were no noise in the system. Another important difference with the perceptron is that all the units in a Boltzmann machine are symmetrically connected; this allows an "energy" to be defined for the network and ensures that the network will relax to an equilibrium state that minimizes the energy (Hopfield 1982). Smolensky (1983, 1986) has studied the same architecture using "harmony," which is the negative of energy, as the global function.

The Boltzmann machine has been applied to a number of constraint satisfaction problems in vision, such as figure-ground separation in image analysis (Sejnowski and Hinton 1986; Kienker et al. 1986), and generalizations have been applied to image restoration (Geman and Geman 1984) and binocular depth perception (Divko and Schulten 1986). Riley and Smolensky (1984) have used harmony theory to study problem solving. The number of times that the network must be updated to reach an optimal solution can be very large when the units are stochastic; an alternative architecture that converges more quickly, although not necessarily to the optimal solution, is based on continuous-valued units (Hopfield 1984; Hopfield and Tank 1985, 1986). This deterministic system is like a "mean field" approximation to the stochastic system.

Boltzmann machines have an interesting learning algorithm that allows "energy landscapes" to be created through training by example. Learning in a Boltzmann machine has two phases. In the training phase a binary input pattern is imposed on the input group and on the correct binary output pattern. The system is allowed to relax to equilibrium at a fixed "temperature" while the inputs and outputs are held fixed. At equilibrium the average fraction of the time a pair of units is on together, the co-occurrence probability p_{ij}^{+}, is computed for each connection. In the test phase the same procedure is followed with only the input units clamped, and the average co-occurrence probabilities p_{ij}^{-} are again computed. The weights are then updated according to

$$\Delta w_{ij} = \varepsilon(p_{ij}^{+} - p_{ij}^{-}), \tag{4.11}$$

where the parameter ε controls the rate of learning. A co-occurrence probability is related to the correlation between the firing or activation of the presynaptic and postsynaptic units and can be implemented by a Hebb synapse. In the second phase, however, the change in the synaptic strengths is anti-Hebbian because it must decrease with increasing correlation. Notice that this procedure is also error correcting, for no change will be made to the weight if the two probabilities are the same. The perceptron learning procedure follows as a special case of the Boltzmann learning algorithm when there are no hidden units and the probability function reduces to a step function.

The Boltzmann learning algorithm has been applied to a variety of problems, such as bandwidth compression (Ackley et al. 1985), the learning of symmetry groups (Sejnowski et al. 1986), and speech recognition (Prager et al. 1986). One of the practical limitations of simulating a Boltzmann machine on a conventional digital computer is the excessive time required to come to equilibrium and collect statistics. A special-purpose VLSI chip is being designed to speed up the learning (Alspector and Allen 1986).

Back-Propagation Another error-correcting learning procedure, introduced by Rumelhart et al. (1986) and called error back-propagation, generalizes the Widrow-Hoff algorithm. The network is a multilayered feed-forward architecture that uses the same processing units described in equation (4.7) and figure 4.1. There may be direct connections between the input layer and the output layer as well as through the hidden units. A superscript is used to denote the layer for each unit, so that $s_i^{(n)}$ is the ith unit on the nth layer. The final output layer is designated the Nth layer.

The first step is to compute the output of the network for a given input. The goal of the learning procedure is to minimize the average squared error between the computed values of the output units and the correct pattern s_i^* provided by a teacher:

$$\text{Error} = \sum_{i=1}^{J} (s_i^* - s_i^{(N)})^2, \tag{4.12}$$

where J is the number of units in the output layer. This is accomplished by first computing the error gradient on the output layer,

$$\delta_i^{(N)} = (s_i^* - s_i^{(N)})P'(E_i^{(N)}), \tag{4.13}$$

and then propagating it backward through the network layer by layer:

$$\delta_i^{(n)} = \sum_j \delta_j^{(n+1)} w_{ji}^{(n)} P'(E_i^{(n)}), \tag{4.14}$$

where $P'(E_i)$ is the first derivative of the function $P(E_i)$ in figure 4.1.

These gradients are the directions that each weight should be altered to reduce the error for a particular item. To reduce the average error for all the input patterns, the gradients must be averaged over all the training patterns before updating the weights. In practice, it is sufficient to average over several inputs before updating the weights. Another method is to compute a running average of the gradient with an exponentially decaying filter:

$$\Delta w_{ij}^{(n)}(u+1) = \alpha \Delta w_{ij}^{(n)}(u) + (1-\alpha)\delta_i^{(n+1)} s_j^{(n)}, \tag{4.15}$$

where α is a smoothing parameter (typically 0.9) and u is the number of input patterns presented. The smoothed weight gradients $\Delta w_{ij}^{(n)}(u)$ can then be used to update the weights:

$$w_{ij}^{(n)}(t+1) = w_{ij}^{(n)}(t) + \varepsilon \Delta w_{ij}^{(n)}, \tag{4.16}$$

where t is the number of weight updates and ε is the learning rate (typically 1.0). The error signal is back-propagated only when the difference between the actual and the desired values of the outputs is greater than a margin of 0.1. This ensures that the network does not overlearn on inputs that it is already getting correct. This learning algorithm can be generalized to networks with feedback connections and multiplicative connections (Rumelhart et al. 1986), but these extensions will not be discussed further.

The definitions of the learning parameters here are somewhat different from those in Rumelhart et al. (1986). In the original algorithm ε is used rather than $(1 - \alpha)$ in equation (4.15). Our parameter α is used to smooth the gradient in a way that is independent of the learning rate ε, which appears only in the weight update [equation (4.16)]. Our averaging procedure also makes it unnecessary to scale the learning rate by the number of presentations per weight update.

The back-propagation learning algorithm has been applied to several problems, including knowledge representation in semantic networks (Hinton 1986; Rumelhart 1986), bandwidth compression by dimensionality reduction (Saund 1986; Zipser 1986), speech recognition (Ellman and Zipser 1986; Watrous et al. 1986), conversion of text to speech (Sejnowski and Rosenberg 1987), and backgammon (Tesauro and Sejnowski 1988). In the next section we give a detailed description of how back-propagation was applied to the problem of converting English text to speech.

Biological Plausibility Neither the Boltzmann machine nor the error back-propagation scheme is meant as a literal model of real neural circuitry. They are also quite different from each other—the Boltzmann machine uses binary stochastic units in a symmetric network, whereas back-propagation uses real-valued deterministic units in a feed-forward network—but both architectures have learning algorithms that depend on gradient descent in the space of weights, which can have high dimensionality. The class of gradient descent algorithms for learning in large networks may have general properties that are already present in the simplest members. Other more elaborate gradient descent learning algorithms, which are more biologically plausible, are also being explored (Parker 1986; Le Cun 1985).

The network models we review in this section make a number of assumptions that should be critically examined. First, the networks are based on a highly idealized version of real neurons, and many constraints concerning patterns of connectivity found in the nervous system are not incorporated into the models. Second, human learning is often imitative rather than instructive, so that children, for

example, are exposed to many positive examples and are not always corrected when they make mistakes.

At this early stage in exploring the capabilities of network models at the psychological level, it is more helpful to discover the general properties of networks before studying the properties of highly specialized networks. The network models are sufficiently general that they can be applied to several different levels of investigation. A processing unit, for example, can be identified with a group of neurons rather than a single neuron and the activity level of the unit identified with the average firing rate within the group. Also, the "teacher" in a supervised learning algorithm should not be taken too literally. For example, one brain area can serve as the teacher and provide the information needed to train another brain area. In the next section we consider the problem of pronouncing English text. The teacher can be a part of the brain that already contains the correct pronunciation of words, and during the learning process the pronunciations become associated with the spellings of the words. Little is known about the neurophysiological basis of human language abilities, so a detailed comparison with real brain circuits is not yet possible.

Thus the present network model should not be considered a neural model but rather a model system in which to explore issues of representation and learning in large populations of neurons. The general insights that are found can be used to explore more detailed brain models and may even help in analyzing recordings from neurons in the cerebral cortex.

4.2 *NETtalk*

The problem of pronouncing written English text illustrates many of the features of skill acquisition and expert performance. In reading aloud, we first recognize letters and words from images on our retinas. Several words can be processed in one fixation so that a significant amount of parallel processing must be involved. At some point in the central nervous system the information encoded visually is transformed into articulatory information about how to produce the correct speech sounds. Finally, intricate patterns of activity occur in the motor neurons that innervate muscles in the larynx and mouth, and sounds are produced. The key step that we are concerned with in this section is the transformation between the highest sensory representations of the letters and the earliest articulatory representations of the phonemes.

English pronunciation has been extensively studied by linguists, and much is known about the correspondence between letters and phonemes (Venezky 1970). English is a particularly difficult language to master because of its irregular orthography. For example, the "a" in almost all words ending in "ave," such as "brave" and "gave," is a long vowel, but not in "have," and there are some words such as "read" that can vary in pronunciation. The problem of reconciling rules and exceptions in converting text to speech shares some characteristics with difficult problems in artificial intelligence that have traditionally been approached with rule-based knowledge representations, such as natural language translation (Haas 1970).

In this section we describe a network that learns to pronounce English text. The model, which we call NETtalk, demonstrates that even a small network can capture a significant fraction of the regularities in English pronunciation as well as absorb many of the irregularities. In commercial systems such as DECtalk (Digital Equipment Corporation), a look-up table (of about a million bits) is used to store the phonetic transcription of the most common words, and phonological rules are applied to words that are not in the dictionary (Allen 1987; Klatt 1980). The result is a string of phonemes that can then be converted to sounds with digital speech synthesis. NETtalk is designed to perform the task of converting strings of letters to strings of phonemes. Earlier work on NETtalk was described by Sejnowski and Rosenberg (1986, 1987).

Network Architecture

NETtalk is a feed-forward network that uses the back-propagation learning algorithm. We have also used the Boltzmann learning algorithm on this problem, but the results are not reported here. The network is hierarchically arranged into three layers of units: an input layer, an output layer, and an intermediate, or "hidden," layer, as illustrated in figure 4.2. Information flows through the network from bottom to top. First, the letter units at the base are clamped; then the states of the hidden units are determined by equations (4.2) and (4.3); finally, the states of the phoneme units at the top are determined.

Representations of Letters and Phonemes The standard network has seven groups of units in the input layer, and one group of units in each of the other two layers. Each input group encodes one letter of the input text, so that strings of seven letters are presented to the input units at any one time. The desired output of the network is the correct phoneme, associated with the center, or fourth, letter of this seven-letter "window." The other six letters (three on either side of

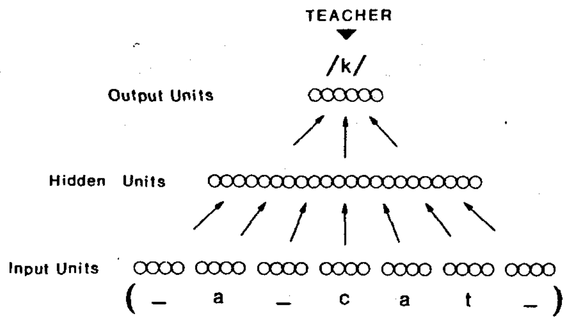

Figure 4.2
Schematic drawing of the NETtalk network architecture. Window of letters in an English text is fed to an array of 203 input units, as shown on the bottom of the pyramid, with 7 groups of 29 units in each group. Information from these units is transformed by an intermediate layer of 80 hidden units. Each hidden unit receives inputs from all the input units on the bottom layer and in turn sends its output to all 26 units in the output layer. The output pattern of activity is then used to choose the closest phoneme and stress corresponding to the middle letter. During learning, a teacher provides the correct output vector and the error is used to update the weights in the network. An example of an input string of letters from a training text is shown below the input groups, and the output phoneme for the middle letter is shown above the output layer. There are 309 units and 18,629 weights in the network, including a variable threshold for each unit.

the center letter) provide a partial context for this decision. The text is stepped through the window letter by letter. At each step the network computes a phoneme, and after each word the weights are adjusted according to how closely the computed pronunciation matched the correct one.

We chose a window with seven letters for two reasons. First, Lucassen and Mercer (1984) have shown that a significant amount of the information needed to pronounce a letter correctly is contributed by the nearby letters (figure 4.3). Second, we were limited by our computational resources to exploring small networks, and it proved possible to train a network with a seven-letter window in a few days. The limited size of the window also meant that some important nonlocal information about pronunciation and stress could not be properly taken into account by our model (Church 1985). The main goal of our model is to explore the basic principles of distributed information coding in a real-world domain rather than to achieve perfect performance.

The letters and phonemes are represented in different ways. The letters are represented locally within each group by twenty-nine dedicated units, one for each letter of the alphabet plus an additional three

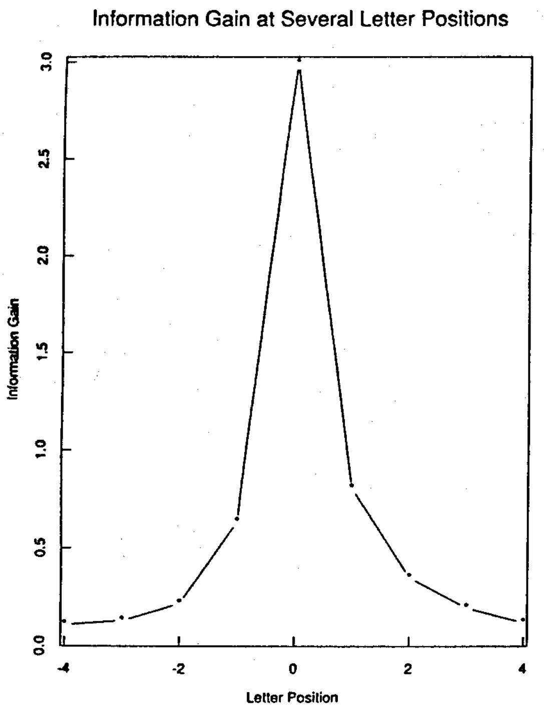

Figure 4.3
Mutual information provided by neighboring letters and the correct pronunciation of the center letter as a function of distance from the center letter. Data from Lucassen and Mercer (1984).

units to encode punctuation and word boundaries. Only one unit in each input group is active for a given input. The phonemes, in contrast, are represented in terms of twenty-one articulatory features, such as point of articulation, voicing, and vowel height, as summarized in table 4.1. Five additional units encode stress and syllable boundaries, making twenty-six output units. This is a distributed representation because each output unit participates in the encoding of several phonemes (Hinton et al. 1986).

The hidden units neither receive direct input nor have direct output but are used by the network to form internal representations appropriate for mapping letters to phonemes. The goal of the learning algorithm is to search effectively the space of all possible weights for a network that performs the mapping.

Learning We used two texts to train the network: phonetic transcriptions from the informal continuous speech of a child (Carterette

Table 4.1
Articulatory representation of phonemes and punctuations

Symbol[a]	Phoneme[b]	Articulatory features[c]
/a/	father	Low, Tensed, Central2
/b/	bet	Voiced, Labial, Stop
/c/	bought	Medium, Velar
/d/	debt	Voiced, Alveolar, Stop
/e/	bake	Medium, Tensed, Front2
/f/	fin	Unvoiced, Labial, Fricative
/g/	guess	Voiced, Velar, Stop
/h/	head	Unvoiced, Glottal, Glide
/i/	Pete	High, Tensed, Front1
/k/	Ken	Unvoiced, Velar, Stop
/l/	let	Voiced, Dental, Liquid
/m/	met	Voiced, Labial, Nasal
/n/	net	Voiced, Alveolar, Nasal
/o/	boat	Medium, Tensed, Back2
/p/	pet	Unvoiced, Labial, Stop
/r/	red	Voiced, Palatal, Liquid
/s/	sit	Unvoiced, Alveolar, Fricative
/t/	test	Unvoiced, Alveolar, Stop
/u/	lute	High, Tensed, Back2
/v/	vest	Voiced, Labial, Fricative
/w/	wet	Voiced, Labial, Glide
/x/	about	Medium, Central2
/y/	yet	Voiced, Palatal, Glide
/z/	zoo	Voiced, Alveolar, Fricative
/A/	bite	Medium, Tensed, Front2 + Central1
/C/	chin	Unvoiced, Palatal, Affricative
/D/	this	Voiced, Dental, Fricative
/E/	bet	Medium, Front1 + Front2
/G/	sing	Voiced, Velar, Nasal
/I/	bit	High, Front1
/J/	gin	Voiced, Velar, Nasal
/K/	sexual	Unvoiced, Palatal, Fricative + Velar, Affricative (Compound: /k/ + /S/)
/L/	bottle	Voiced, Alveolar, Liquid
/M/	absym	Voiced, Dental, Nasal
/N/	button	Voiced, Palatal, Nasal
/O/	boy	Medium, Tensed, Central1 + Central2
/Q/	quest	Voiced, Labial + Velar, Affricative, Stop
/R/	bird	Voiced, Velar, Liquid
/S/	shin	Unvoiced, Palatal, Fricative

Table 4.1 *(Continued)*

Symbol[a]	Phoneme[b]	Articulatory features[c]
/T/	*th*in	Unvoiced, Dental, Fricative
/U/	book	High, Back1
/W/	bout	High + Medium, Tensed, Central2 + Back1
/X/	excess	Unvoiced, Affricative, Front2 + Central1 (Compound: /k/ + /s/)
/Y/	cute	High, Tensed, Front1 + Front2 + Central1
/Z/	leisure	Voiced, Palatal, Fricative
/@/	b*a*t	Low, Front2
/!/	Nazi	Unvoiced, Labial + Dental, Affricative (Compound: /t/ + /s/)
/#/	examine	Voiced, Palatal, + Velar, Affricative (Compound: /g/ + /z/)
/*/	one	Voiced, Glide, Front1 + Low, Central1 (Compound: /w/ + /^/)
/|/	log*i*c	High, Front1 + Front2
/^/	b*u*t	Low, Central1
/-/	Continuation	Silent, Elide
/–/	Word boundary	Pause, Elide
/./	Period	Pause, Full Stop
<	Syllable boundary	Right
>	Syllable boundary	Left
1	Primary stress	Strong, weak
2	Secondary stress	Strong
0	Tertiary stress	Weak
–	Word boundary	Right, left, boundary

a. The symbols for phonemes are a superset of ARPAbet and are associated with the sound of the italicized part of the adjacent word.
b. Compound phonemes were introduced when a single letter was associated with more than one primary phoneme.
c. Two or more of the following twenty-one articulatory feature units were used to represent each phoneme and punctuation. Position in mouth: Labial = Front1, Dental = Front2, Alveolar = Central1, Palatal = Central2, Velar = Back1, Glottal = Back2. Phoneme type: Stop, Nasal, Fricative, Affricative, Glide, Liquid, Voiced, Tensed. Vowel height: High, Medium, Low. Punctuation: Silent, Elide, Pause, Full stop. The continuation symbol was used when a letter is silent. Stress and syllable boundaries were represented with combinations of five additional units, as shown at the end of this table. Stress was associated with vowels, and arrows were associated with letters. The arrows point toward the stress and change direction at syllable boundaries. Thus the stress assignments for "atmosphere" are 1 < > 0 >>> 2 <<. The phoneme and stress assignments were chosen independently.

and Jones 1974) and Merriam-Webster's *Pocket Dictionary* (1974). The corresponding letters and phonemes were aligned, and a special symbol for continuation, -, was inserted whenever a letter is silent or part of a graphemic letter combination, as in the conversion from the string of letters "phone" to the string of phonemes /f-on-/ (see table 4.1). Two procedures were used to move the text through the window of seven input groups. For the corpus of informal continuous speech the text was processed in order with word boundary symbols between the words. Several words or word fragments could be within the window at the same time. For the dictionary the words were placed in random order and moved through the window individually.

The weights were incrementally adjusted during the training according to the discrepancy between the desired and the actual values of the output units. For each phoneme this error was "back-propagated" from the output to the input layer using the learning algorithm introduced by Rumelhart et al. (1986) and described in the previous section. Each weight in the network was adjusted after every word to minimize its contribution to the total mean squared error between the desired and the actual output. The weights in the network were always initialized to small random values uniformly distributed between −0.3 and 0.3; this was necessary to differentiate the hidden units.

A simulator was written in the C programming language for configuring a network with arbitrary connectivity, training it on a corpus and collecting statistics on its performance. A network of 10,000 weights had a throughput during learning of about 2 letters/sec on a VAX 780 FPA. After every presentation of an input the inner product of the output vector was computed with the codes for each of the phonemes. The phoneme that made the smallest angle with the output was chosen as the "best guess." Slightly better performance was achieved by choosing the phoneme whose representation had the smallest Euclidean distance from the output vector, but these results are not reported here. All performance figures reported in the next section refer to the percentage of correct phonemes chosen by the network. The performance was also assayed by "playing" the output string of phonemes and stresses through DECtalk, bypassing the part of the machine that converts letters to phonemes.

Performance

Continuous Informal Speech Carterette and Jones (1974) provide phonetic transcriptions of children and adults that were taped during

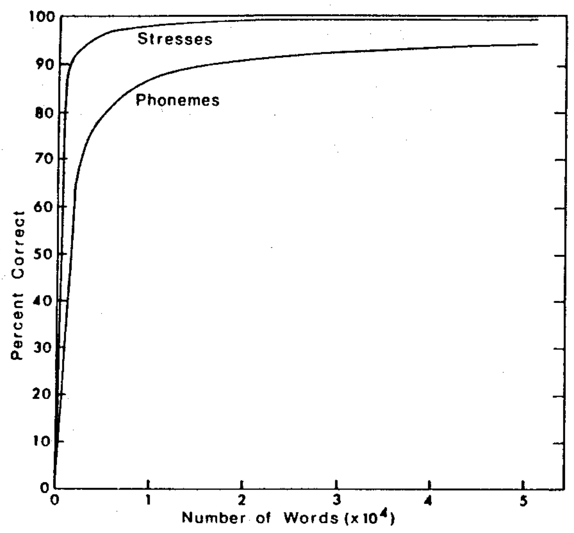

Figure 4.4
Learning curves for phonemes and stresses during training on the 1,024-word corpus of continuous informal speech. The percentage of correct phonemes and stresses are shown as functions of the number of training words.

informal sessions. This was a particularly difficult training corpus because the same word was often pronounced several different ways; phonemes were commonly elided or modified at word boundaries, and adults were about as inconsistent as children. We used the first two pages of transcriptions, which contain 1,024 words from a child in the first grade. The stresses were assigned to the transcriptions so that the training text sounded natural when played through DECtalk. The learning curve for 1,024 words from the informal speech corpus is shown in figure 4.4. The percentage of correct phonemes rises rapidly at first and continues to rise at a slower rate throughout the learning, reaching 95% after 50,000 words. Primary and secondary stresses and syllable boundaries are learned quickly for all words, and the network achieved nearly perfect performance by 5,000 words. When the learning curves are plotted on double logarithmic scales, they are approximately straight lines; thus the learning follows a power law, which is characteristic of human skill learning (Rosenbloom and Newell 1986).

The distinction between vowels and consonants is made early; however, the network predicts the same vowel for all vowels and the same consonant for all consonants, which results in a babbling sound. A second stage occurs when word boundaries are recognized, and the output then resembles pseudowords. After just a few passes

through the network many of the words are intelligible, and by ten passes the text is understandable.

When the network makes an error, it often substitutes phonemes that sound similar. For example, a common confusion is between the "th" sounds in "thesis" and "these," which differ only in voicing. Few errors in a well-trained network are confusions between vowels and consonants. Some errors are actually corrections to inconsistencies in the original training corpus. Overall the intelligibility of the speech is quite good.

Does the network memorize the training words, or does it capture the regular features of pronunciation? As a test of generalization, a network trained on the 1,024-word corpus of informal speech was tested without training on a 439-word continuation from the same speaker. The performance was 78%, which indicates that much of the learning was transferred to novel words even after a small sample of English words.

Is the network resistant to damage? We examined performance of a highly trained network after making random changes of varying size to the weights. As shown in figure 4.5a, random perturbations of the weights uniformly distributed on the interval $[-0.5, 0.5]$ have little effect on the performance of the network, and degradation is gradual with increasing damage. This damage causes the average magnitude of each weight to change by 0.25; this can be considered the roundoff error that can be tolerated before the performance of the network begins to deteriorate. With 4 binary bits it is possible to specify 16 possible values, or -2 to $+2$ in steps of 0.25. This range covers almost all the weights, which have an average magnitude of 0.8. Hence the minimum information needed to specify each weight in the network is about 4 bits.

If the damage is not too severe, relearning is much faster than the original learning starting from the same level of performance, as shown in figure 4.5b. Similar fault tolerance and fast recovery from damage has also been observed in networks constructed using the Boltzmann learning algorithm (Hinton and Sejnowski 1986).

Dictionary The Merriam-Webster *Pocket Dictionary* we used has 20,012 words. A subset of the 1,000 most commonly occurring words was selected from this dictionary based on frequency counts in the Brown corpus (Kuchera and Francis 1967). The most common English words are also among the most irregular, so this was also a test of the capacity of the network to absorb exceptions. We were particularly interested in exploring how the performance of the network and

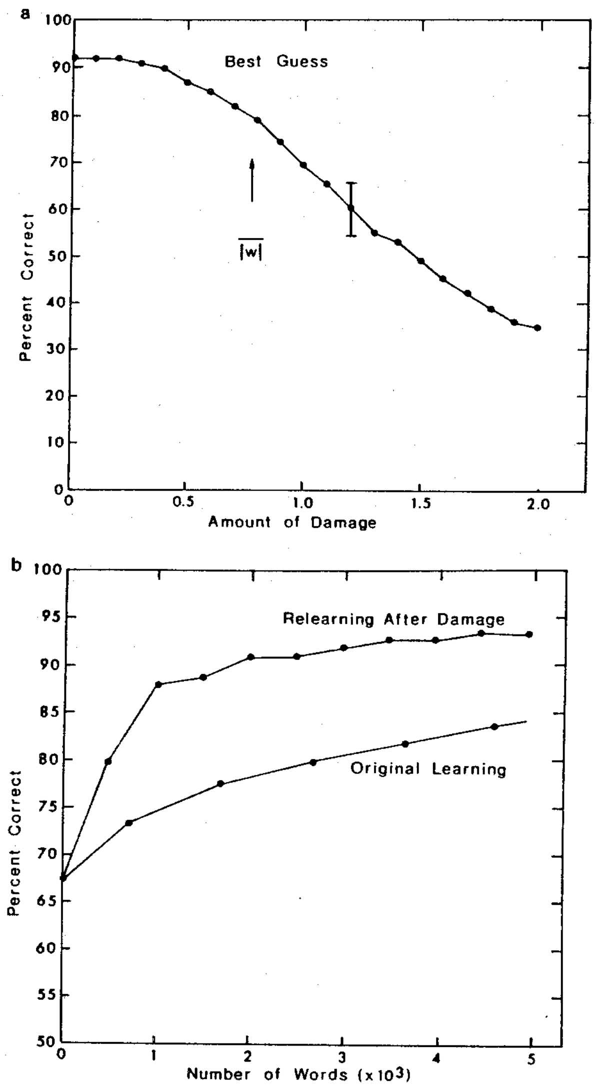

Figure 4.5
Damage to the network and recovery from damage. (a) Performance of a network as a function of the amount of damage to the weights. The network had been previously trained on 50 passes through the corpus of continuous informal speech. The weights were then damaged by adding a random component to each weight uniformly distributed on the interval $[-d, d]$, where d is the amount of damage plotted on the abscissa. The performance shown is the average of at least two disrupted networks for each value of d. For $d = 1.2$, 22 disrupted networks were tested to obtain a standard deviation of 6%. The average absolute value of the weights in the network was $|w| = 0.77$, and the standard deviation was $\sigma = 1.2$. There was little degradation of the best guesses until $d = 0.5$, and the falloff with increasing damage was gentle. (b) Retraining of a damaged network compared with the original learning curve starting from the same level of performance. The network was damaged with $d = 1.2$ and was retrained using the same corpus and learning parameters that were used to train it. There is a rapid recovery phase during the first pass through the network, followed by a slower healing process similar in time course to the later stages of the original training. These two phases can be accounted for by the shape of the error metric in weight space, which typically has deep ravines (Hinton and Sejnowski 1986).

learning rate scale with the number of hidden units. With no hidden units the performance rises quickly and saturates at 82%, as shown in figure 4.6a. This represents the part of the mapping that can be accomplished by linearly separable partitioning of the input space (Minsky and Papert 1969). Hidden units allow more contextual influence by recognizing higher-order features among combinations of input units.

The rate of learning and asymptotic performance increases with the number of hidden units, as shown in figure 4.6a. The best performance achieved with 120 hidden units was 98%, significantly better than the performance achieved with continuous informal speech, which was more difficult because of the variability in real-world speech. Different letter-to-sound correspondences are learned at different rates; two examples are shown in figure 4.6b. The ability of a network to generalize was tested on a large dictionary. Using weights from a network with 120 hidden units trained on the 1,000 words, the average performance of the network on the dictionary of 20,012 words was 77%. With continued learning the performance reached 85% at the end of the first pass through the dictionary, indicating a significant improvement in generalization. Following five training passes through the dictionary, the performance increased to 90%.

The number of input groups was varied from three to eleven. Both the speed of learning and the asymptotic level of performance improved with the size of the window. The learning curve with 11 input groups and 80 hidden units was about 7% higher than a network with 7 input groups and 80 hidden units up to about 25,000 words of training and reached 97.5% at 55,000 words, compared with 95% for the network with 7 input groups.

Adding an extra layer of hidden units also improved the performance somewhat. A network with 7 input groups and two layers of 80 hidden units each was trained first on the 1,000-word dictionary. Its performance after 55,000 words of training was 97%, and its generalization was 80% on the 20,012-word dictionary without additional training and 87% after the first pass through the dictionary with

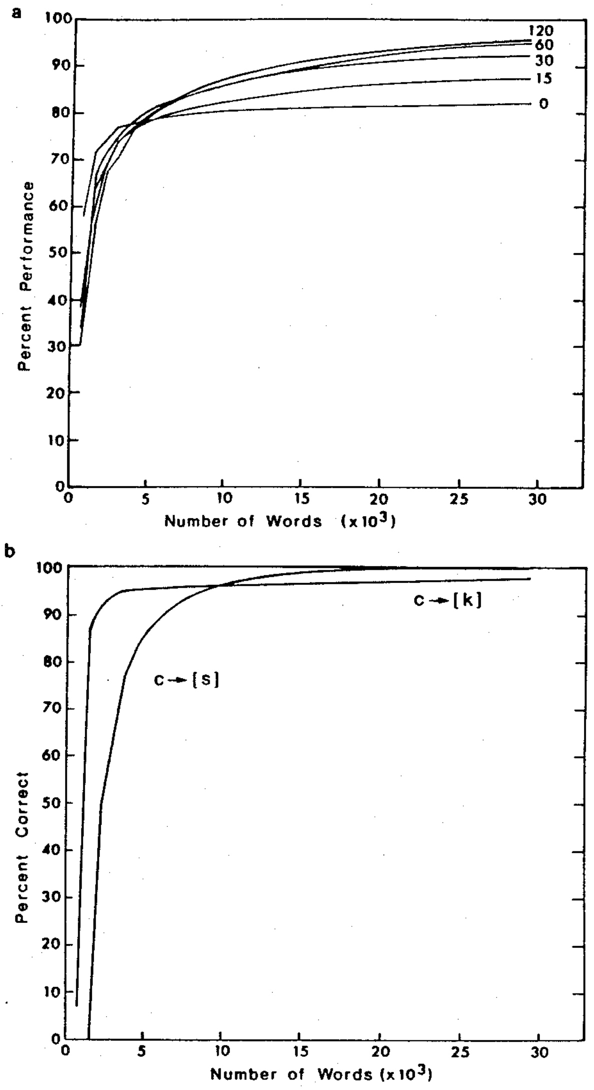

Figure 4.6
(a) Learning curves for training on a corpus of the 1,000 most common words in English using different numbers of hidden units, as indicated for each curve. The percentage of phonemes correctly assigned by the network is shown as a function of the number of training words. For the case with no hidden units, the input units were directly connected to the output units. (b) Performance during learning of two representative phonological rules, the hard and soft pronunciation of the letter "c." Note that the soft "c" takes longer to learn but eventually achieves perfect accuracy. The hard "c" occurs about twice as often as the soft "c" in the training corpus. Children show a similar difficulty with learning to read words with the soft "c" (Venezky and Johnson, 1973).

training. The asymptotic performance after 218,000 words of training on the dictionary was 91%. Compared to the network with 120 hidden units, which had about the same number of weights, the network with two layers of hidden units was better at generalization but about the same in absolute performance.

Analysis of Hidden Units

There are not enough hidden units in even the largest network that we studied to memorize all the words in the dictionary. The standard network with 80 hidden units had 18,629 weights, including variable thresholds. If we allow 4 bits of accuracy for each weight, as indicated by the damage experiments, the total storage needed to define the network is about 10 kilobytes or 80,000 bits. In comparison, the 20,012-word dictionary, including stress information, requires nearly 2,000,000 bits of storage. This data compression is possible because of the redundancy in English pronunciation. By studying the patterns of activation among the hidden units, we were able to understand some of the coding methods that the network had discovered.

The standard network used for analysis had 7 input groups and 80 hidden units and had been trained to 95% correct on 1,000 dictionary words. The levels of activation of the hidden units were examined for each letter of each word using the graphical representation shown in figure 4.7. On average, about 20% of the hidden units are highly activated for any given input, and most of the remaining hidden units have little or no activation. Thus the coding scheme cannot be described as either a local representation, which would activate only one or two units, or a "holographic" representation, in which all the hidden units participate to some extent. It is apparent, even without using statistical techniques, that many hidden units are highly activated only for certain letters or sounds or letter-to-sound correspondences. Some of the hidden units can be assigned unequivocal characterizations, such as one unit that responds only to vowels, but most of the units participate in more than one regularity.

To test the hypothesis that letter-to-sound correspondences are the primary organizing variable, we computed the average activation level of each hidden unit for each letter-to-sound correspondence in the training corpus. The result was 79 vectors with 80 components each, one vector for each letter-to-sound correspondence. A hierarchical clustering technique was used to arrange the letter-to-sound vectors in groups based on a Euclidean metric in the 80-dimensional space of hidden units. The overall pattern, as shown in figure 4.8, is striking: The most important distinction is the complete separation of consonants and vowels. However, within these two groups the clus-

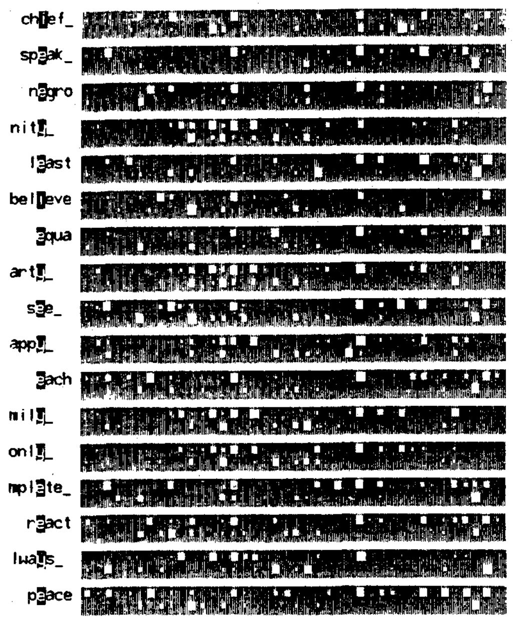

Figure 4.7
Levels of activation in the layer of hidden units for a variety of words, all of which produce the same phoneme, /E/, on the output. The network had 7 input groups and 80 hidden units. The input string is shown on the left with the center letter emphasized. The level of activity of each hidden unit is shown on the right, in two rows of 40 units each. The area of the square is proportional to the activity level. Only a few units were highly activated, and most were inactive.

tering has a different pattern. For the vowels the next most important variable is the letter, whereas consonants are clustered according to the similarity of their sounds. The same clustering procedure is repeated for three networks starting from different random starting states. The patterns of weights are completely different, but the clustering analysis reveals the same hierarchies, with some differences in the details, for all three networks.

4.3 *The Spacing Effect*

In section 4.2 we demonstrated that a small network was able to perform a difficult task in a way that was quite different from most previous methods used to solve the problem. Information about particular words are stored in the network in a distributed fashion. New words can be added to the network, but they must be added in a way that is compatible with the previously stored information. As a consequence, the representation of information affects the training strategy for learning new words. In this section we compare the performance of humans on learning paired associates with the performance of the network on a similar task in the domain of pronunciation. More details about these simulations are presented by Rosenberg and Sejnowski (1986).

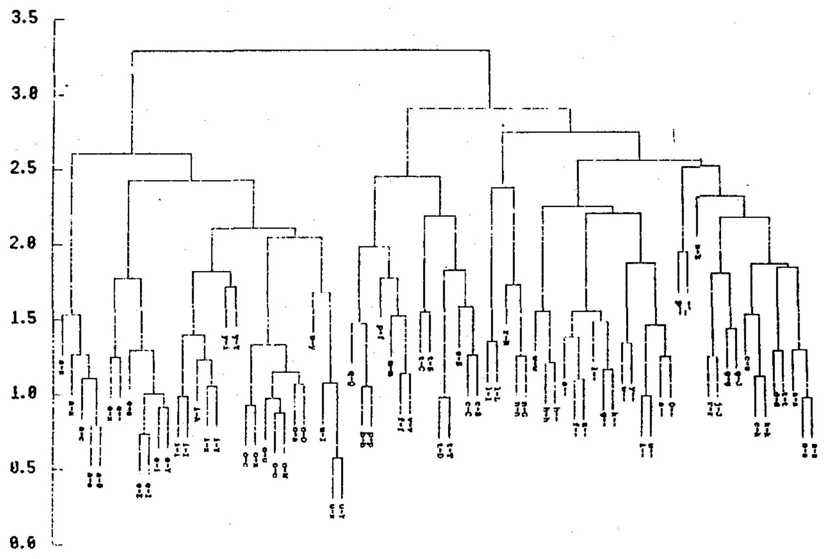

Figure 4.8
Hierarchical clustering of hidden units for letter-to-sound correspondences. The vectors of average hidden unit activity for each correspondence, shown at the bottom of the binary tree, were sequentially grouped according to an agglomerative method using complete linkage (Everitt 1974). The horizontal scale gives the Euclidean distance between the farthest elements in two groups when a pair was merged.

Spaced versus Massed Practice

In 1885 Ebbinghaus noted that "with any considerable number of repetitions a suitable distribution of them over a space of time is decidedly more advantageous than the massing of them at a single time" (Ebbinghaus 1885, p. 89). Since then, the spacing effect has been found across a wide range of stimulus materials and tasks, semantic as well as perceptual and motor, and has even been found when the repetitions are across modality, or across languages if bilinguals are employed as subjects [see Hintzman (1974) for a review]. The ubiquity of these results suggests that spacing reflects something of central importance in memory. However, despite over a hundred years of research, there is no adequate, or at least simple, explanation for the spacing effect.

Perhaps the most popular account of the spacing effect is the encoding variability hypothesis, which assumes that stimuli are encoded relative to the context, or environment, in which they occur and that the probability of recall is greater when the context at retrieval is similar to the context at encoding. Another explanation is that the subject habituates to repeated presentations of the same item and therefore cannot process the later presentations as well as the first. Jacoby (1978) has suggested that less conscious "processing effort" is made by the subjects to subsequent presentations when they are massed rather than spaced. Although each of these explanations can account for some of the experimental data, none can account for all experiments (Hintzman 1976).

These theories attempt to explain spacing in terms of such concepts as encoding, habituation, and consolidation, which make little reference to the actual form of the memory representation. Another approach is to seek an explanation at the level of the representation: It may matter how the information is stored in the system. One way to explore this possibility is to construct explicit models that incorporate particular memory representations and learning mechanisms and to test them with the same experimental paradigms that have been used to study human memory.

In this section we demonstrate that the spacing effect also occurs in NETtalk when the same experimental paradigm used to study the spacing effect on humans is applied to the network described in section 4.2. The window size was reduced from seven to five to speed training. There were 231 units and 10,346 connections in the version of the network used in the present experiments.

Experimental Design

The design was modeled after Glenburg's Experiment 1 (Glenberg 1976). In this experiment subjects were presented with paired associates, repeated twice at spacings of approximately 0, 1, 4, 8, 20, and 40 intervening items, and tested at retention intervals of approximately 2, 8, 32, and 64 items. Each pair was composed of two four-letter common nouns "constructed to avoid common preexperimental associations, rhymes, and orthographic similarities" (table 4.2). During testing only the stimulus word was presented, and the subject was to recall the associated response term. Glenberg's results are reproduced here as figure 4.9. A significant interaction is found between spacing (lag) and the retention interval. At short retention intervals massed repetitions lead to a higher probability of recall, whereas at long retention intervals distributed repetitions are advantageous. Glenberg also noted that retention at the 64-item re-

Table 4.2
Examples of some training distractors and target items used in the experiments on spacing in NETtalk[a]

Letters	Phonemes	Stress
Distractors[a]		
file	fAl-	>1<<
all	cl-	1<<
second	sEkxnd	>1<0<<
take	tek-	>1<<
together	txgED--R	>0>1<<0<
neck	nEk-	>1<<
atmosphere	@tmxsf-Ir-	1<>0>>>2<<
Random target items		
fozepd	WdicnK	1<121>
sccfyk	p-UdSp	>202<1
bmyqcl	bzgTlz	0>><<>
grtufh	KCczOL	>1<010
eqhxxu	ANT|vM	>01<>2
ncssvr	zTSdWg	<<12>2
wxsale	RKpfl|	1<1110
djzxde	Yby^yI	20>>2>
kmfjqi	WGenGN	1><102

a. Training distractors were part of the original training corpus and were presented between training sessions on the target items and during the retention interval.

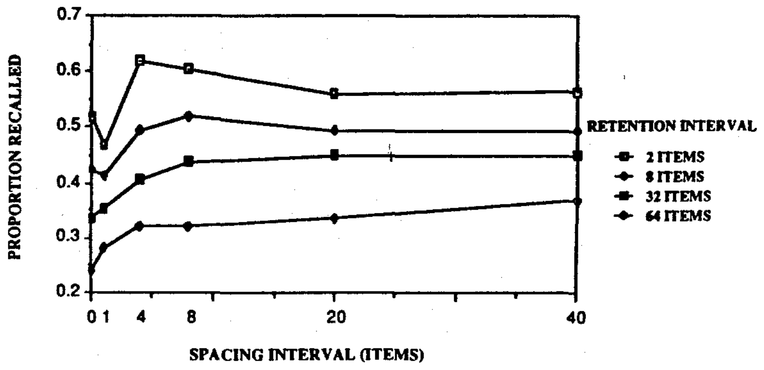

Figure 4.9
The proportion of response terms recalled as a function of spacing interval and retention interval. After Glenberg (1976).

tention interval is a monotonic and negatively accelerating function of spacing.

As in Glenberg's experiment, we measured the retention of target stimuli repeated a certain number of times at various spacing intervals as a function of the retention interval. If NETtalk exhibits the spacing effect, then long-term retention of these items should be better when a large number of other items intervene between successive repeats of the target (distributed practice). Conversely short-term retention of the target items should be better when fewer items are presented between repeats (massed practice).

Pre-Experimental Training The network was first trained on the 1,000 most common English words taken from the Brown corpus. The network cycled through this 1,000-word corpus eleven times. The performance of the network at this point in training, as determined by the percentage of the correct phonemes "guessed," was 85% and could have been improved with further practice. The weight values of the network were stored following this initial training and served as a common starting point for all of the subsequent experimental trials.

Target Stimuli In order to force new learning to take place, random character strings of length six were employed as target stimuli. Thus there was no orderly relation between the cue and the response. Whatever performance level NETtalk was able to reach on these items could not have been due to the utilization of rules acquired either before or after the study. Twenty six-letter cues were generated by choosing six letters at random (with replacement) out of the twenty-six letters of the English alphabet. Likewise, the response terms associated with each of these cues were randomly generated phoneme and stress strings, also six characters in length. There were fifty-three possible phonemes and five possible stress characters. In generating the target stimuli, two "phonemes," the space between words (_) and the period (.), were not possible choices. The frequency of occurrence of the characters in natural language were not taken into account in this selection process. Some of these items and several items from the training corpus are presented in table 4.2.

Procedure The 20 target items were tested individually on separate trials. A trial consisted of, first, reading in the pre-experimental weights, presenting a target item 2, 10, or 20 times, and then measuring the retention of the target as it was interfered with by subsequent learning. Furthermore, each target was presented at each of 6 spacing intervals, with either 0 (massed), 1, 4, 8, 20, or 40 (distributed) intervening items. Thus 18 trials were devoted to each target item (3 repetition groups × 6 spacing intervals). Between successive repeats of the target, words were presented from the original training corpus. Following the last repeat, the training corpus was again presented, and retention of the response terms of the target item was assessed after every item by presenting the cue term and measuring the mean squared difference between the output of the network and the correct response. The error defined in equation (4.12) was used to define the response accuracy for the word:

$$\text{Accuracy} = 1 - \frac{\sum_{l=1}^{L} \text{Error}_l}{L}, \tag{4.17}$$

where L is the number of letters in the word. Note that this measure of accuracy is more sensitive than the performance accuracy given earlier, which measured only the correct choices made by the network. Learning was turned off (achieved by setting the learning rate to 0) for these tests, so that no changes were made to the strengths of the connections in the network.

Results Accuracy, as defined by equation (4.17) was averaged over the 20 target items and plotted as a function of spacing interval for each repetition group at retention intervals of 2, 8, 32, and 64 in figure 4.10, following Glenberg (1976). A significant spacing effect was observed in NETtalk: Retention of nonwords after a 64-item retention interval was significantly better when presented at the longer spacings (distributed presentation) than at the shorter spacings. In addition, a significant advantage for massed presentations was found for short-term retention of the items. Although stimulus materials, response measures, and procedure differ sufficiently to make direct comparison impossible, the overall pattern of these results resembles that found by Glenberg (1976) in an experiment using human subjects. We obtained our results without making additional assumptions or including additional mechanisms such as consolidation, rehearsal, or attention. Nor were explicit assumptions made about a continuously changing context other than the context implicitly provided by the network.

Recency effects, similar to those reported here, are common in the literature and have been reported in many spacing experiments [for example, Peterson et al. (1963) and Sperber (1974)]. This short-term advantage for massed practice is commonly discussed with reference to a limited-capacity memory buffer. The present experiments indi-

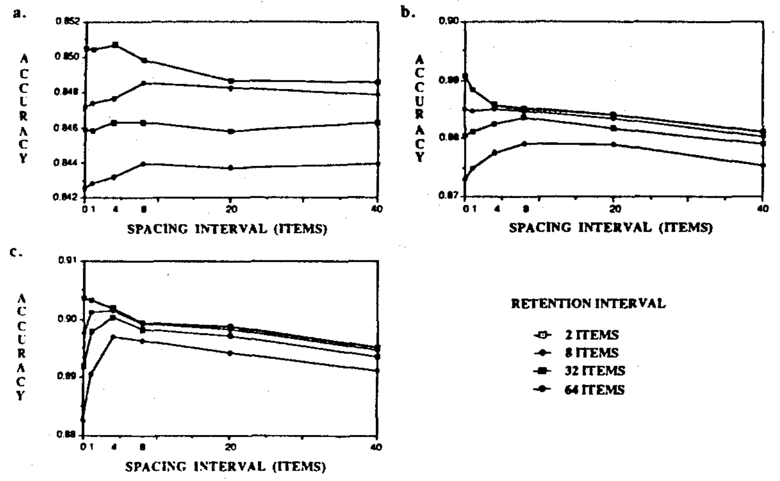

Figure 4.10
Mean response accuracy plotted as a function of spacing interval at 2-, 8-, 32-, and 64-item retention intervals for the (a) 2, (b) 10, and (c) 20 repetition groups.

cate that some of the effects for which such a mechanism is designed to account can be produced without a separate buffer.

A Possible Explanation for the Spacing Effect

Why should NETtalk exhibit these characteristics? The answer depends on the learning procedure and the way in which the resulting knowledge is represented in the network.

The learning algorithm alters the weights by a small amount after each training word in a direction that minimizes the average error. A network with *n* weights can be considered a point in the *n*-dimensional Euclidean space of weights, and this point moves through the space during learning along a trajectory that brings it closer to a point in the space where the error over the entire training corpus is minimal. After reaching such a point, the network is stable; that is, further training on the same vocabulary will not change the weights. If a new word that is irregular is introduced, then the network must accommodate the new word in such a way that the pronunciations of the old words are not altered. Our hypothesis is that distributing practice leads to a more stable position in the weight space upon the re-presentation of the training corpus.

For the sake of simplicity, consider only three connections from the entire network, so that they can be represented in a three-dimensional space (see figure 4.11). Suppose further that, as in the

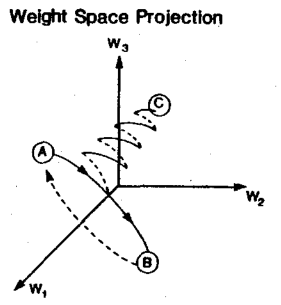

Figure 4.11
Idealized trajectories in weight space during learning for massed (dark) and distributed (light) conditions. Point A is a set of weights that is optimum for the pre-experimental training corpus (the assumed starting point for all experimental trials); point B is an optimum for the target item, and point C is an optimum for both the target and the training corpus. See text for explanation.

present simulations, this network has been trained on a large pre-experimental training corpus and that it has reached a point where the error is at a local minimum for these items (point A). Now a new and unusual target item is presented in either a massed or spaced condition to the network. If the target is presented several times back to back, as in the massed condition, minimizing the error following each presentation leads the weights down a path toward a nearby point that is optimal for this target item, perhaps even reaching it (point B). But because this trajectory will have taken the network some distance away from the starting point, this new position is not likely to be stable to the re-presentation of the training corpus, and so the massed learning of the new item will be lost quickly.

Assuming, however, that there is a point that is optimal for both the training corpus and the target item (point C in figure 4.11), alternating presentations of the target with items from the training set is one way of moving closer to this highly stable point. On the first presentation of the target item the error gradient for that item is estimated and the error is reduced by adjusting the weights in the direction of the steepest descent (to position 1). So far, this procedure has been identical with that for the massed condition, so the network is at the same point in weight space. Now, however, instead of presenting the target again, an item from the original training corpus is presented. Again the weights are adjusted to minimize the error on the item (to position 2), only this time the direction of movement is more likely to be toward point A than point B, because A is a global minimum for the training corpus. Presenting the target again will cause a movement back toward B (to position 3), and so on. We see that distributing practice causes the network to weave through the weight space, allowing it to search for a point that is good for both the training corpus and the target item. The network therefore has a better chance of finding the overall optimal position (point C) than it would if practice were massed, and its encoding of the target item will consequently be more able to withstand interference resulting from further training on both types of material.

Discussion

In all the experiments we updated the weights after every word. Another way to learn new items is to update the weight values less frequently. Instead of learning in small increments, one could also collect data over many trials and then take one big jump. Although this procedure (within its resolution) overcomes the problems associated with presentation order (such as the spacing effect), it may be hazardous because new information is integrated at a slow rate. Both of these time scales for modification might be used in the nervous system. Hinton (personal communication) has suggested that each synapse could have one component that changes its value rapidly and another component that changes more gradually. Fast learning could be done with the fast component of the weight, and only an average of the fast synaptic changes could be committed to long-term storage. This allows new regions of the weight space to be temporarily explored without "forgetting" the previous knowledge.

The explanation of the spacing effect that we offer here is not meant as an alternative to previous suggestions; it is a different type of explanation, relying as it does on the underlying structure of the representations. The decline in learning rate as local optima are approached is reminiscent of the process of habituation: Less is effectively learned each time the item is repeated. Other aspects of our model bear a resemblance to encoding variability to the extent that items are encoded relative to the current state of the network, which is in a state of continual flux. And if we identify Jacoby's processing effort with the degree of change required to construct a distributed representation, then our simulations can be considered support for this proposal as well. Nevertheless, although these concepts of habituation, encoding variability, and processing effort may be reinterpreted within the framework of connectionist models such as ours, they are at a different level of explanation.

Our results are limited to a particular network architecture in a particular domain. To what extent is this conclusion dependent on the details of our model? If the spacing effect is a direct consequence of incremental learning in memory systems that use distributed rep-

resentations, as we suspect, then the same effects of massed and distributed learning should occur in other task domains and with other network architectures that also have learning algorithms with distributed representations, such as Boltzmann machines (Hinton and Sejnowski, 1983; Ackley et al. 1985). We predict as well that the same general principles may underlie the spacing effect in human learning.

4.4 Conclusions

NETtalk is an illustration in miniature of many aspects of learning. First, the network starts with considerable "innate" knowledge of input and output representations chosen by the experimenters and with no knowledge specific for English—the network could have been trained on any language with the same set of letters and phonemes. Second, the network acquires its competence through practice, goes through several distinct stages, and reaches a significant level of performance. Finally, the information is distributed in the network such that no single unit or link is essential. As a consequence, the network is fault tolerant and degrades gracefully with increasing damage. Moreover, the network recovers from damage much more quickly than it takes to learn initially. In addition to these features, the effect of temporal ordering during training on new words is remarkably similar to that in humans.

Despite these similarities with human learning and memory, NETtalk is too simple to serve as a good model for the acquisition of reading skills in humans. The network attempts to accomplish in one stage what occurs in two stages of human development. Children learn to talk first, and only after representations for words and their meanings are well developed do they learn to read. It is also likely that we have access to articulatory representations for whole words in addition to our ability to use letter-to-sound correspondences, but there are no word-level representations in the network. It is perhaps surprising that the network is capable of reaching a significant level of performance using a window of only seven letters. This approach would have to be generalized to account for prosodic features in connected text, and a human level of performance would require the integration of information from several words at once.

NETtalk can be used as a research tool to explore many aspects of network coding, scaling, and training in a domain that is far from trivial. Those aspects of the network's performance that are similar to human performance are good candidates for general properties of network models; more progress may be made by studying these aspects in the small test laboratory that NETtalk affords. Our exploration of the spacing effect is an example of how a general property of human memory can be studied in a much simpler model system. When NETtalk deviates from human performance, there is good reason to believe that a more detailed account of brain circuitry may be necessary.

After training many networks, we concluded that many different sets of weights give about equally good performance. Although it was possible to understand the function of some hidden units, it was not possible to identify units in different networks that have the same function. However, the activity patterns in the hidden units could be interpreted in an interesting way. Patterns of activity in groups of hidden units could be identified in different networks that serve the same function, such as distinguishing vowels from consonants. This suggests that the detailed synaptic connectivity between neurons in the cerebral cortex may not be helpful in revealing the functional properties of a neural network. It is not at the level of the synapse or the neuron that one should expect to find invariant properties of a network but at the level of functional groupings of cells. Techniques that are developed to uncover these groupings in model neural networks could be of value in uncovering similar cell assemblies in real neural networks.

Acknowledgments

We thank Alfonso Caramazza, Francis Crick, Stephen Hanson, Geoffrey Hinton, Thomas Landauer, James McClelland, George Miller, David Rumelhart, and Stephen Wolfram for helpful discussions about language and learning. We are indebted to Stephen Hanson and Andrew Olson, who made important contributions in the statistical analysis of the hidden units, and to Thomas Landauer, who suggested that the spacing effect could be studied with NETtalk. Peter Brown, Edward Carterette, Howard Nusbaum, and Alex Waibel assisted in the early stages of development. Bell Communications Research generously provided computational support.

Terrence J. Sejnowski is supported by grants from the National Science Foundation, the System Development Foundation, the Sloan Foundation, General Electric Corporation, the Allied Corporation Foundation, the Richard Lounsbery Foundation, the Seaver Institute, and the Air Force Office of Scientific Research. Charles R. Rosenberg is supported in part by grants from the James S. McDonnell Foundation, by IBM under grant 487906, by the Defense Advanced Research Projects Agency of the Department of Defense, by the Office of Naval Research under contracts N00014-85-C-0456 and N00014-85-K-0465, and by the National Science Foundation under Cooperative Agreement DCR-8420948 and grant IST8503968.

References

Ackley, D. H., G. E. Hinton, and T. J. Sejnowski. 1985. "A learning algorithm for Boltzmann machines." *Cognitive Science* 9:147–169.

Adrian, E. D. 1953. *The Physical Background of Perception*. Oxford: Clarendon Press.

Allen, J. 1987. *From Text to Speech: the MITalk System*. Cambridge: Cambridge University Press.

Allman, J. M., J. F. Baker, W. T. Newsome, and S. E. Petersen. 1983. "Visual topography and function: Cortical visual areas in the owl monkey," in *Cortical Sensory Organization. Vol. 2, Multiple Visual Areas*, C. N. Woolsey, ed. Clifton, N.J.: Humana Press.

Alspector, J., and R. B. Allen. 1986. *A VLSI Model of Neural Nets*. Technical Memorandum TM ARH002688. Morristown, N.J.: Bellcore.

Anderson, J. A. 1970. "Two models for memory organization using interacting traces." *Mathematical Biosciences* 8:137–160.

Anderson, J. A., and M. C. Mozer. 1981. "Categorization and selective neurons," in *Parallel Models of Associative Memory*, G. E. Hinton and J. A. Anderson, eds. Hillsdale, N.J.: Erlbaum Associates.

Anderson, J. R. 1982. "Acquisition of cognitive skill." *Psychological Review* 89:369–406.

Arbib, M. A. 1987. *Brains, Machines and Mathematics*, second edition. New York: McGraw-Hill.

Ballard, D. H., G. E. Hinton, and T. J. Sejnowski. 1983. "Parallel visual computation." *Nature* 306:21–26.

Barlow, H. B. 1972. "Single units and sensation: A neuron doctrine for perceptual psychology?" *Perception* 1:371–394.

Barto, A. G. 1985. "Learning by statistical cooperation of self-interested neuronlike computing elements." *Human Neurobiology* 4:229–256.

Baum, E. B., J. Moody, and F. Wilczek. 1987. *Internal Representations for Associative Memory*. Technical Report. Santa Barbara, Cal.: Institute for Theoretical Physics, University of California.

Bienenstock, E. L., L. N. Cooper, and P. W. Munro. 1982. "Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex." *Journal of Neuroscience* 2:32–48.

Bounds, D. G. 1986. "Numerical simulations of Boltzmann machines," in *Neural Networks for Computing* (AIP Conference Proceedings), J. S. Denker, ed. New York: American Institute of Physics, vol. 151, 59–64.

Carterette, E. C., and M. G. Jones. 1974. *Informal Speech*. Los Angeles, Calif.: University of California Press.

Chauvet, G. 1986. "Habituation rules for a theory of the cerebellar cortex." *Biological Cybernetics* 55:201–209.

Church, K. 1985. "Stress assignment in letter to sound rules for speech synthesis," in *Proceedings of the 23rd Annual Meeting of the Association for Computational Linguistics*, D. Walker, ed. Morristown, NJ: ACL, 246–253.

Churchland, P. S. 1986. *Neurophilosophy*. Cambridge, Mass.: MIT Press.

Cohen, M. A., and S. Grossberg. 1983. "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks." *IEEE Transactions on Systems, Man and Cybernetics* 13:815–825.

Cooper, L. N., F. Liberman, and E. Oja. 1979. "A theory for the acquisition and loss of neuron specificity in visual cortex." *Biological Cybernetics* 33:9–28.

Crick, F. H. C., and C. Asanuma. 1986. "Certain aspects of the anatomy and physiology of the cerebral cortex," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press, 333–371.

Divko, R., and K. Schulten. 1986. "Stochastic spin models for pattern recognition," in *Neural Networks for Computing* (AIP Conference Proceedings), J. S. Denker, ed. New York: American Institute of Physics, vol. 151, 129–134.

Ebbinghaus, H. 1885. *Memory: A Contribution to Experimental Psychology*. Berlin: Privat Docent in Philosophy at the University of Berlin. Reprinted in 1964 by Dover (New York).

Ellman, J., and D. Zipser. 1987. *Learning the Hidden Structure of Speech*. Technical Report 8701. San Diego, Cal.: University of California, Institute for Cognitive Science.

Everitt, B. 1974. *Cluster Analysis*. London: Heinemann.

Fahlman, S. E., G. E. Hinton, and T. J. Sejnowski. 1983. "Massively parallel architectures for AI: NETL, THISTLE and Boltzmann machines." *Proceedings of the National Conference on Artificial Intelligence*. Los Altos, Cal.: William Kauffman, Inc., 109–113.

Feldman, J. A. 1982. "Dynamic connections in neural networks." *Biological Cybernetics* 46:27–39.

Feldman, J. A. 1986. *Neural Representation of Conceptual Knowledge*. Technical Report TR-189. Rochester, N.Y.: University of Rochester, Department of Computer Science.

Feldman, J. A., and D. H. Ballard. 1982. "Connectionist models and their properties." *Cognitive Science* 6:205–254.

Finkel, L. H., and G. M. Edelman. 1985. "Interaction of synaptic modification rules within population of neurons." *Proceedings of the National Academy of Sciences USA* 82:1291–1295.

Gamba, A. L., G. Gamberini, G. Palmieri, and R. Sanna. 1961. "Further experiments with PAPA." *Nuovo Cimento*, suppl., 20(2):221–231.

Geman, S., and D. Geman. 1984. "Stochastic relaxation, Gibbs distributions, and the Baysian restoration of images." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 3:79–92.

Glenberg, A. M. 1976. "Monotonic and nonmonotonic lag effects in paired-associate and recognition memory paradigms." *Journal of Verbal Learning and Verbal Behavior* 15:1–16.

Gluck, M. A., and G. H. Bower. 1986. "Conditioning and categorization: Some common effects of informational variables in animal and human learning." *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, N.J.: Erlbaum Associates, 126–140.

Gluck, M. A., and G. H. Bower. 1987. "From conditioning to category learning: An adaptive network model." Unpublished.

Gluck, M. A., and R. F. Thompson. 1986. "Modeling the neural substrates of associative learning and memory: A computational approach." *Psychological Review* 94(2):176–191.

Grossberg, S. 1976. "Adaptive pattern classification and universal recoding I. Parallel development and coding of neural feature detectors." *Biological Cybernetics* 23:121–134.

Haas, W. 1970. *Phonographic Translation*. Manchester: Manchester University Press.

Hebb, D. O. 1949. *Organization of Behavior*. New York: Wiley.

Hinton, G. E. 1986. "Learning distributed representations of concepts." *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, N.J.: Erlbaum, 1–12.

Hinton, G. E., and J. A. Anderson. 1981. *Parallel Models of Associative Memory*. Hillsdale, N.J.: Erlbaum Associates.

Hinton, G. E., and T. J. Sejnowski. 1983. "Optimal perceptual inference." *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Silver Spring, Md.: IEEE Computer Society Press, 448–453.

Hinton, G. E., and T. J. Sejnowski. 1986. "Learning and relearning in Boltzmann machines," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press, 282–317.

Hinton, G. E., J. L. McClelland, and D. E. Rumelhart. 1986. "Distributed representations," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1, Foundations*, D. E. Rumelhart and J. L. McClelland, eds. Cambridge, Mass.: MIT Press, 77–109.

Hintzman, D. L. 1974. "Theoretical implications of the spacing effect," in *Theories in Cognitive Psychology: The Loyola Symposium*, R. L. Solso, ed. Hillsdale, N.J.: Erlbaum Associates, 77–99.

Hintzman, D. L. 1976. "Repetition and memory," in *The Psychology of Learning and Motivation*, G. H. Bower, ed. New York: Academic Press, vol. 10, 47–91.

Hopfield, J. J. 1982. "Neural networks and physical systems with emergent collective computational abilities." *Proceedings of the National Academy of Sciences USA* 79:2554–2558.

Hopfield, J. J. 1984. "Neurons with graded response have collective computation abilities." *Proceedings of the National Academy of Sciences USA* 81:3088–3092.

Hopfield, J. J., and D. Tank. 1985. "Neural computation of decision in optimization problems." *Biological Cybernetics* 52:141–152.

Hopfield, J. J., and D. Tank. 1986. "Computing with neural circuits: A model." *Science* 233:624–633.

Hubel, D. H., and T. N. Wiesel. 1962. "Receptive fields, binocular interactions, and functional architecture in the cat's visual cortex." *Journal of Physiology* 160:106–154.

Jacoby, L. L. 1978. "On interpreting the effects of repetition: Solving a problem versus remembering a solution." *Journal of Verbal Learning and Verbal Behavior* 17:649–667.

Kahneman, D., and A. Tversky. 1972. "Subjective probability: A judgment of representativeness." *Cognitive Psychology* 3:430–454.

Kelso, S. R., A. H. Ganong, and T. H. Brown. 1986. "Hebbian synapses in hippocampus." *Proceedings of the National Academy of Sciences USA* 83:5326–5330.

Kienker, P. K., T. J. Sejnowski, G. E. Hinton, and L. E. Schumacher. 1986. "Separating figure from ground with a parallel network." *Perception* 15:197–216.

Klatt, D. 1980. "Software for a cascade/parallel formant synthesizer." *Journal of the Acoustical Society of America* 67:971–995.

Klopf, A. H. 1986. "A drive-reinforcement model of single neuron function: An alternative to the Hebbian neuronal model, in *Neural Networks for Computing*, J. S. Denker, ed. New York: American Institute of Physics, 265–270.

Kohonen, T. 1970. "Correlation matrix memories." *IEEE Transactions on Computers* C21:353–359.

Kohonen, T. 1984. *Self-Organization and Associative Memory*. New York: Springer-Verlag.

Kolmogorov, A. N. 1957. "On the representation of continuous functions of one variable by superposition of continuous functions of one variable and addition." *AMS Translation* 2:55–59.

Kuchera, H., and W. N. Francis. 1967. *Computational Analysis of Modern-Day American English*. Providence, R.I.: Brown University Press.

Le Cun, Y. 1985. "A learning procedure for asymmetric network." *Proceedings of Cognitiva* 85:599–604.

Levy, W. B., J. A. Anderson, and W. Lehmkuhle. 1984. *Synaptic Change in the Nervous System*. Hillsdale, N.J.: Erlbaum Associates.

Levy, W. B., S. E. Brassel, and S. D. Moore. 1983. "Partial quantification of the associative synaptic learning rule of the dentate gyrus." *Neuroscience* 8:799–808.

Linsker, R. 1986. "From basic network principles to neural architecture: Emergence of orientation columns." *Proceedings of the National Academy of Sciences USA* 83:8779–8783.

Longuet-Higgins, H. C. 1968. "Holographic model of temporal recall." *Nature* 217:104–107.

Lucassen, J. M., and R. L. Mercer. 1984. "An information theoretic approach to the automatic determination of phonemic baseforms." *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*. Silver Spring, Md.: IEEE Press, 42.5.1–42.5.4.

Lynch, G. 1986. *Synapses, Circuits, and the Beginnings of Memory*. Cambridge, Mass.: MIT Press.

Lyons, J. 1971. *Introduction to Theoretical Linguistics*. Cambridge: Cambridge University Press.

Marr, D., and T. Poggio. 1976. "Cooperative computation of stereo disparity." *Science* 194:283–287.

McClelland, J. L., and D. E. Rumelhart. 1986. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*. Cambridge, Mass.: MIT Press.

McCulloch, W. S., and W. H. Pitts. 1943. "A logical calculus of ideas immanent in nervous activity." *Bulletin of Mathematical Biophysics* 5:115–133.

McNaughton, B. L., and R. G. Morris. 1987. "Hippocampal synaptic enhancement and information storage within a distributed memory system." *Trends in Neuroscience* 10:408–415.

Minsky, M., and S. Papert. 1969. *Perceptrons*. Cambridge, Mass.: MIT Press.

Mountcastle, V. B. 1978. "An organizing principle for cerebral function: The unit module and the distributed system," in *The Mindful Brain*, G. M. Edelman and V. B. Mountcastle, eds. Cambridge, Mass.: MIT Press, 7–50.

Norman, D. A. 1982. *Learning and Memory*. San Francisco, Calif.: Freeman.

Palm, G. 1978. "On representation and approximation on nonlinear systems." *Biological Cybernetics* 31:119–124.

Palm, G. 1979. "On representation and approximation of nonlinear systems II. Discrete time." *Biological Cybernetics* 34:49–52.

Parker, D. B. 1986. "A comparison of algorithms for neuron-like cells," in *Neural Networks for Computing*, J. S. Denker, ed. New York: American Institute of Physics, 327–332.

Pearlmutter, B. A., and G. E. Hinton. 1986. "G-Maximization: An unsupervised learning procedure for discovering regularities," in *Neural Networks for Computing*, J. S. Denker, ed. New York: American Institute of Physics, 333–338.

Peterson, L. R., R. Wampler, M. Kirkpatrick, and D. Saltzman. 1963. "Effect of spacing presentations on retention of a paired-associate over short intervals." *Journal of Experimental Psychology* 66:206–209.

Prager, R. W., T. D. Harrison, and F. Fallside. 1986. *Boltzmann Machines for Speech Recognition*. Technical Report TR.260. Cambridge: Cambridge University, Engineering Department.

Reggia, J. A. 1985. "Virtual lateral inhibition in parallel activation models of associative memory." *Proceedings of the 9th International Joint Conference on Artificial Intelligence*. Los Altos, Calif.: Morgan Kauffman, 244–248.

Rescorla, R. A., and A. R. Wagner. 1972. "A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement," in *Classical Conditioning II: Current Research and Theory*, A. H. Black and W. F. Prokasy, eds. New York: Appleton-Crofts, 64–99.

Riley, M. S., and P. Smolensky. 1984. "A parallel model of (sequential) problem solving." *Proceedings of the Sixth Annual Conference of the Cognitive Science Society*. Hillsdale, N.J.: Erlbaum Associates, 286–292.

Rolls, E. T. 1986. "Information representation, processing and storage in the brain: Analysis at the single neuron level," in *Neural and Molecular Mechanisms of Learning*. Berlin: Springer-Verlag.

Rosenberg, C. R., and T. J. Sejnowski. 1986. "The spacing effect on NETtalk, a massively-parallel network." *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*. Hillsdale, N.J.: Erlbaum Associates, 72–89.

Rosenblatt, F. 1959. *Principles of Neurodynamics*. New York: Spartan Books.

Rosenbloom, P. S., and A. Newell. 1986. "The chunking of goal hierarchies: A generalized model of practice," in *Machine Learning: An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, eds. Los Altos, Calif.: Morgan Kauffman, vol. 2, 247–288.

Rumelhart, D. E. 1986. "Learning paradigms in connectionism." Paper presented at the Symposium on Connectionism: Multiple Agents, Parallelism and Learning. Geneva, Switzerland, September 9–12, 1986.

Rumelhart, D. E., and J. L. McClelland. 1986a. "On learning the past tenses of English verbs." in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press.

Rumelhart, D. E., and J. L. McClelland. 1986b. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1, Foundations*. Cambridge, Mass.: MIT Press.

Rumelhart, D. E., and D. Zipser. 1985. "Feature discovery by competitive learning." *Cognitive Science* 9:75–112.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams. 1986. "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1, Foundations*, D. E. Rumelhart and J. L. McClelland, eds. Cambridge, Mass.: MIT Press.

Saund, E. 1986. "Abstraction and representation of continuous variables in connectionist networks." *Proceedings of the Fifth National Conference on Artificial Intelligence*. Los Altos, Calif.: Morgan Kauffmann, 638–644.

Sejnowski, T. J. 1977a. "Statistical constraints on synaptic plasticity." *Journal of Mathematical Biology* 69:385–389.

Sejnowski, T. J. 1977b. "Storing covariance with nonlinearly interacting neurons." *Journal of Mathematical Biology* 4:303–321.

Sejnowski, T. J. 1981. "Skeleton filters in the brain," in *Parallel Models of Associative Memory*, G. E. Hinton and J. A. Anderson, eds. Hillsdale, N.J.: Erlbaum Associates, 189–212.

Sejnowski, T. J. 1986. "Open questions about computation in cerebral cortex," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press, 372–389.

Sejnowski, T. J., and G. E. Hinton. 1987. "Separating figure from ground with a Boltzmann machine," in *Vision, Brain and Cooperative Computation*, M. A. Arbib and A. R. Hanson, eds. Cambridge, Mass.: MIT Press.

Sejnowski, T. J., and C. R. Rosenberg. 1986. *NETtalk: A Parallel Network That Learns to Read Aloud*. Technical Report 86/01. Baltimore, Md.: Johns Hopkins University, Department of Electrical Engineering and Computer Science

Sejnowski, T. J., and C. R. Rosenberg, 1987. "Parallel networks that learn to pronounce English text." *Complex Systems* 1:145–168.

Sejnowski, T. J., P. K. Kienker, and G. E. Hinton. 1986. "Learning symmetry groups with hidden units: Beyond the perceptron." *Physica* 22D:260–275.

Smolensky, P. 1983. "Schema selection and stochastic inference in modular environments." in *Proceedings of the National Conference on Artificial Intelligence*. Los Altos, California: William Kauffman, 378–382.

Smolensky, P. 1986. "Information processing in dynamical systems: Foundations of harmony theory," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 2, Psychological and Biological Models*, J. L. McClelland and D. E. Rumelhart, eds. Cambridge, Mass.: MIT Press, 194–281.

Sperber, R. D. 1974. "Developmental changes in effects of spacing of trials in retardate discrimination learning and memory." *Journal of Experimental Psychology* 103:204–210.

Squire, L. R. 1986. "Mechanisms of memory." *Science* 232:1612–1619.

Steinbuch, K. 1961. "Die lernmatrix." *Kybernetik* 1:36–45.

Sutton, R. S., and A. G. Barto. 1981. "Toward a modern theory of adaptive networks: Expectation and prediction." *Psychological Review* 88:135–170.

Tesauro, G. 1986. "Simple neural models of classical conditioning." *Biological Cybernetics* 55:187–200.

Tesauro, G., and T. J. Sejnowski. In press. "A parallel network that learns to play backgammon." *Artificial Intelligence Journal*.

Toulouse, G., S. Dehaene, and J.-P. Changeux. 1986. "Spin glass model of learning by selection." *Proceedings of the National Academy of Sciences USA* 83:1695–1698.

Tulving, E. 1985. "How many memory systems are there?" *American Psychologist* 40(4):385–398.

Van Essen, D. C., and J. H. R. Maunsell. 1983. "Hierarchical organization and functional streams in the visual cortex." *Trends in Neuroscience* 6:370–375.

Venezky, R. L. 1970. *The Structure of English Orthography*. The Hague: Mouton.

Venezky, R. L., and D. Johnson. 1973. "Development of two letter-sound patterns in grades one through three." *Journal of Educational Psychology* 64:109–115.

von der Malsburg, C., and E. Bienenstock. 1987. "A neural network for the retrieval of superimposed connection patterns." *Europhysics Letters* 3:1243–1249.

Watrous, R. L., and L. Shastri. 1986. *Learning Phonetic Features Using Connectionist Networks: An Experiment in Speech Recognition*. Technical Report MS-CIS-86-78. Philadelphia, Penn.: University of Pennsylvania Department of Electrical Engineering and Computer Science.

Widrow, G., and M. E. Hoff. 1960. "Adaptive switching circuits." Institute of Radio Engineers Western Electronic Show and Convention. *Convention Record*, vol. 4, 96–194.

Willshaw, D. 1981. "Holography, associative memory, and inductive generalization," in *Parallel Models of Associative Memory*, G. E. Hinton and J. A. Anderson, eds. Hillsdale, N.J.: Erlbaum Associates, 83–104.

Zipser, D. 1986. *Programming Neural Nets to Do Spatial Computations*. ICS Technical Report 8608. San Diego, Calif.: University of California, Institute for Cognitive Science.

II

Psychological Dimensions of Memory Function in Humans