Psychological Review
1990, Vol. 97, No. 2, 285-308

Copyright 1990 by the American Psychological Association, Inc.
0033-295X/90/$00.75

Connectionist Models of Recognition Memory: Constraints Imposed by Learning and Forgetting Functions

Roger Ratcliff
Northwestern University

Multilayer connectionist models of memory based on the encoder model using the backpropagation learning rule are evaluated. The models are applied to standard recognition memory procedures in which items are studied sequentially and then tested for retention. Sequential learning in these models leads to 2 major problems. First, well-learned information is forgotten rapidly as new information is learned. Second, discrimination between studied items and new items either decreases or is nonmonotonic as a function of learning. To address these problems, manipulations of the network within the multilayer model and several variants of the multilayer model were examined, including a model with prelearned memory and a context model, but none solved the problems. The problems discussed provide limitations on connectionist models applied to human memory and in tasks where information to be learned is not all available during learning.

The first stage of the connectionist revolution in psychology is reaching maturity and perhaps drawing to an end. This stage has been concerned with the exploration of classes of models, and the criteria that have been used to evaluate the success of an application have been necessarily loose. In the early stages of development of a new approach, lax acceptability criteria are appropriate because of the large range of models to be examined. However, there comes a second stage when the models serve as competitors to existing models developed within other theoretical frameworks, and they have to be competitively evaluated according to more stringent criteria. A few notable connectionist models have reached these standards, whereas others have not. The second stage of development also requires that the connectionist models be evaluated in areas where their potential for success is not immediately obvious. One such area is recognition memory. The work presented in this article evaluates several variants of the multilayer connectionist model as accounts of empirical results in this area. I mainly discuss multilayer models using the error-correcting backpropagation algorithm and do not address other architectures such as adaptive resonance schemes (Carpenter & Grossberg, 1987).

Before launching into the modeling of recognition memory, I need to specify the aims and rules under which this project was carried out. This is important in a new area of inquiry because there are many divergent views about what needs to be done to advance theory. In a systematic application of connectionist models to an empirical domain, examining the whole range (possibly infinite) of potential models is not possible. There is always the suggestion that "if you only try. . . ." This response to the failure of a particular model can be shortsighted. Although a new fix might deal with one difficulty, it will probably not deal with the range of phenomena that need to be explained, unless the implications are thoroughly understood. Of course, a variant that handles the failures I discuss (as well as other empirical phenomena in recognition memory) is the outcome of this work that is most desirable.

The approach I take in this article is to examine the more obvious variants of a class of models and apply them systematically to empirical results in recognition memory. The aim is to provide a cumulative study of the models, examining both their strong and weak points, and to establish recognition memory as an empirical area for further theoretical development using connectionist models. The focus on different variants across classes of models allows general strengths and weaknesses that apply across the classes to be examined, and so this approach makes inappropriate the if-you-only-try response.

An important corollary to this approach concerns the application of the models to domains that are similar or related to those in which the models have previously failed. This kind of application should be viewed with suspicion. For example, a multilayer model that has problems with learning and forgetting in recognition memory should not be applied in related domains (e.g., frequency judgment or categorization) for which recognition can be shown to be a special case unless the previous failures can be adequately addressed.

The presentation in this article includes considerable detail in some sections. For a more rapid reading, summaries or previews are provided for some of the main conclusions of the detailed sections.

Connectionist Models

The neurally inspired connectionist models have recently become major challengers to the more traditional information-

This research was supported by National Science Foundation (NSF) Grant 85-10361 and National Institute of Mental Health Grant MH 44640 to Roger Ratcliff and NSF Grant 85-16350 to Gail McKoon. I thank Gail McKoon, Neal Cohen, Douglas Hintzman, Michael McCloskey and Richard Shiffrin for extremely valuable discussion of the research presented in this article and Douglas Medin, Bennet Murdock, Charles Rosenberg, and two anonymous reviewers for comments on the manuscript.

Correspondence concerning the article should be addressed to Roger Ratcliff, Psychology Department, Northwestern University, Evanston, Illinois 60208.

processing models in cognitive psychology. Although there has been work on such neurally inspired models for several years (e.g., J. A. Anderson, 1972, 1973; J. A. Anderson & Hinton, 1981; J. A. Anderson, Silverstein, Ritz, & Jones, 1977; Grossberg, 1980, 1988; Kohonen, 1978), recent interest has been fueled by the finding that earlier limitations of simple one-layer models, such as their inability to compute various functions (Minsky & Papert, 1969), do not hold when there is more than one layer and when nonlinearity is introduced into processing (Rumelhart & McClelland, 1986). In addition, a body of successful applications of these connectionist architectures to several domains of study in cognitive psychology has been produced.

In this first stage of the connectionist revolution, the emphasis has been (properly) on accounting for major findings within various areas. Individual models have been quite successful, but the success has brought some skepticism. The focus of the skepticism concerns the issue of constraints on the models. No matter what the successes of the models, there is an additional need for information about the limitations of the models and about phenomena that the models cannot explain. One major goal of this article is to use the memory domain to investigate limitations on the multilayer connectionist architecture, and in so doing provide results that will help to develop an understanding of general constraints on the models.

The particular area I address is recognition memory. Recognition memory is an important area of study because it is a fundamental aspect of memory and one that has an extensive data base. Equally important, responses in recognition tests are rapid (for single words, typically 500–800 ms), so that the recognition process is less susceptible to the effects of conscious strategies than other, slower tasks. Thus, it can be argued that the processes underlying recognition memory belong in the category of the *microstructure of cognition,* the term that has been used to describe the basic parallel processing approach to the elementary processes underlying cognition (Rumelhart & McClelland, 1986). Memory is also an area in which there are substantive quantitative models that can be used as benchmarks for these investigations (Gillund & Shiffrin, 1984; Hintzman, 1986; Murdock, 1982; Ratcliff & McKoon, 1988). These global memory models have been developed to explain a range of experimental data across a range of experimental paradigms. The model of Gillund and Shiffrin (1984) was designed to explain both recall and recognition performance as a function of a number of independent variables. Hintzman (1986, 1988) has applied his model to recognition, frequency judgments, and data from categorization procedures. Murdock (1982, 1983), Eich (1982, 1985), and Pike (1984) have applied their models to subsets of data from recognition, serial recall, cued recall, and free recall. All of these models are primarily concerned with explaining patterns of data across experimental paradigms, with a major effort toward capturing quantitative relationships.

The initial aim of the research described in this article was to develop a multilayer connectionist model and apply it to the domain of memory phenomena as a competitor for the global memory models. Besides providing a connectionist account of recognition memory, the model would be useful in providing competitive and contrasting comparisons. However, the more obvious connectionist models failed, as did several variants. Thus, in this article, I document these failures and attempt to explain in detail the properties of the models that led to the failures.

The empirical phenomena to be examined in this article concern learning and forgetting in the standard recognition memory paradigm. In a typical experiment, subjects study a list of words, and then they are given a test list of words for recognition. The test list contains half old words that were on the study list and half new words that were not on the study list. The old and new words are presented in random order, and the subject's task is to decide for each word whether it is old or new. For example, with a study list of 16 words, the test list would contain 32 words, half old and half new. Typical measures of performance are accuracy in terms of proportion correct and discriminability between old items and new items (another important measure is reaction time, but I do not consider it here; see Ratcliff, 1978, 1988). Learning in this context is defined as improvement in discriminability (d') between old and new test items as a function of amount of study. Forgetting is defined as the decrement in accuracy as a function of intervening information presented between study and test of the item.

The first and main model I evaluate is the encoder network (Ackley, Hinton, & Sejnowski, 1985), using the backpropagation learning algorithm. The encoder model was originally devised as a test of learning algorithms; the model reproduces any one of a previously trained set of input vectors at its output layer, with a narrow channel (fewer hidden units than input or output units) between the input and output layers (see Figure 1). As the model is implemented in the work described in this article, items entered into the system are assumed to be vectors of features, typically of length 32 (although some initial demonstrations use vectors of length 4). The system is made up of an input layer of this number of units, a hidden layer of fewer units (½ to ¾ the number of units in the input layer), and an output layer of the same size as the input layer. The hidden units have no direct connections to or from the external input or output. Instead, the hidden units project to units at the input and output layers so that each unit in the hidden layer is connected to each unit in the input layer and also to each unit in the output layer. The task of the system is to reproduce an input signal as an output signal. During learning, an item vector is presented at the input layer, and activation is transmitted through the network, resulting in a set of values at the units of the output layer. The differences between the values of the input and the values at the output layer are used as the training signals to modify the weights in the network. Weights are changed in proportion to the difference between the actual output value and the required output value for a particular unit. In this model, the location of memory can be conceptualized as being distributed across the interconnections between the input and hidden layers and the hidden and output layers.

The tests of the model that follow move from examination of a simple system of four vectors to examination of larger systems for which statistical properties are averaged over large numbers of Monte Carlo simulations. The simple system was studied to examine the behavior of individual elements within a vector as a function of learning and forgetting. The larger systems were studied as potentially viable models of recognition memory in

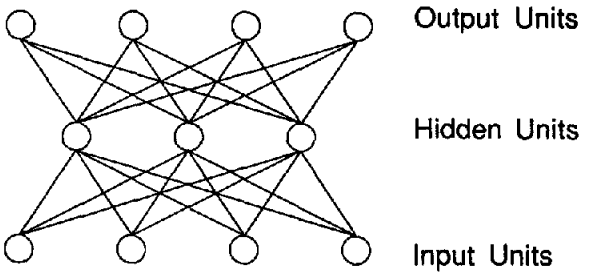

Figure 1. The multilayer encoder model.

which statistical properties of the data could be examined. To model recognition, a set of vectors is trained (the old items), and then at test, these old vectors and some new vectors (randomly chosen with the same statistical properties as the old vectors) are presented to the system. The backpropagation algorithm takes an input vector and produces an output vector with elements that vary between 0 and 1. For the encoder model, the desired output is the same as the input, and to quantify the degree of match between the output and the input, a match value based on the dot product of the two vectors is used. From the match values for tests of old and new vectors and the variability in these values (over the Monte Carlo trials), the discriminability between old and new items is computed.

The input vectors used in the simulations are random vectors with elements either 0 or 1, and this is a potential limitation. In many tasks, information is organized and structured, and issues of representation are of major concern. Some of the problems encountered in the simulations might disappear if reasonable representations were used instead of random vectors. In the evaluation of variants of the encoder model, some representation issues are examined (e.g., learning as increments to prelearned memory, and stimulus vectors as variants on a prototype), but these examples are by no means exhaustive. Until further theoretical development of feature representations, this is probably the best that can be done (see also the representation assumptions made by McCloskey & Cohen, 1989, that produce results similar to those presented later).

A Multilayer Distributed Connectionist Model of Recognition Memory

The multilayer model was chosen for study because it is an architecture that is currently being applied to many different domains in psychology. In addition, it allows computation of functions that linear models (single or multiple layer) are unable to compute (Rumelhart, Hinton, & Williams, 1986), a problem that was used to argue against neural models (Minsky & Papert, 1969). A further reason for choosing the nonlinear backpropagation algorithm in contrast to linear models is that some linear models cannot account for performance in recognition memory because they predict that discrimination between studied items and nonstudied items remains constant as a function of amount of study (J. A. Anderson, 1973; Murdock, 1982).

In the multilayer model, items are represented as vectors, with each feature or element having a value between 0 and 1. An input vector with elements o_j and the weights between the nodes in the input and hidden layers are used to compute the input to the hidden layer:

$$\text{net}_i = \sum_j w_{ij} o_j. \tag{1}$$

This net input to the hidden layer is transformed to activation (the output of the hidden layer) by a logistic sigmoid transformation that transforms net_i values from a range of $-\infty$ to $+\infty$ to the range 0 to 1:

$$o_i = 1/[1 + \exp(-\text{net}_i)]. \tag{2}$$

The activation o_i and the set of weights between the nodes in the hidden and output layers are used to produce the net input to the output layer by Equation 1. This net input is transformed with the logistic transformation to activation at the output layer, which is the final output from the system. Once activation values are obtained for the output layer, they are used in conjunction with the desired or target output values to modify the weights in the system. The weights are modified to minimize the sum of the squared error between the value of each target element and the corresponding element of the obtained output. This is analogous to minimizing the energy in a physical system, where energy is defined as the sum of squared differences between the desired target vector and the output vector produced by the system. With the logistic function (Equation 2), the error signal at an output element is given by

$$\delta_i = (t_i - o_i)o_i(1 - o_i), \tag{3}$$

where t_i is the teaching signal (or desired output) for element i (Rumelhart et al., 1986). Then this error value is used to change the weights between the hidden and output layers with

$$\Delta w_{ij} = \eta \delta_i o_j, \tag{4}$$

where η is a learning rate parameter. Once the weights between the output and hidden layer are modified, the weights from the hidden layer to the input layer have to be modified. The error signal is computed by propagating the error from the output layer to the hidden layer from

$$\delta_i = o_i(1 - o_i) \sum_k w_{ik} \delta_k. \tag{5}$$

The change in the weights between the hidden and input layers is given by Equation 4 with δ_i from Equation 5. Weights were updated after each presentation of a vector to the system during learning but not during testing.

There are two other factors to consider before proceeding to simulations. First, if the sum of squares or energy space is highly variable as a function of parameters and if a high learning rate is used (large η in Equation 4 leading to large changes in weights), the system can oscillate and so produce slow convergence to an adequate solution. To avoid this problem, weight modification is based on a linear combination of the current computed weight change and the prior weight change. The component of weight change that is based on earlier weight changes is termed *momentum* and refers to the fact that prior weight modifications as well as the newly computed changes guide weight modification (see Rumelhart et al., 1986). In many of the simulations presented later, a value of momentum set to 0.5 was used, but to check the generality of the results, the qualita-

tive behavior of the model was compared to the case for which the value of momentum was set to 0 (i.e., weight changes based only on current computation). Results showed few differences (except in the case of mixed vs. pure list presentation; see specific examples presented later).

The second factor to consider is called *symmetry breaking* by Rumelhart et al. (1986), and the term refers to the fact that if weights start with equal values, modifications can only keep the weights equal because the hidden units will all receive the same error signal back from the output layer. To overcome this problem, small random unequal values are assigned as initial values to the weights (e.g., from a uniform distribution in the range −0.3 to +0.3). Again, variation in the size of these initial values was examined, and the effects are noted later in the article. One additional factor sometimes used in the multilayer model is bias on the nodes. This adds a term in the exponent of Equation 2 (see Rumelhart et al., 1986) and is used to capture the average match across all nodes. No biases on the weights were used in these simulations.

Last, a measure of the degree of recognition for a test vector is required. With the encoder model, the appropriate measure is the dot product between the input and output vectors (transformed to the range −1 to +1 and divided by the vector length). The model attempts to reproduce the input at the output, and the dot product measures how well this is accomplished. To form the dot product, each element is transformed to a new value 2 times the old value minus 1; this provides values of the elements in the range −1 to +1. The dot product between the transformed vectors is then divided by the number of elements to provide an overall value of match between −1 and +1. With several Monte Carlo replications, the mean in the match value and its standard deviation can be obtained for both old and new items and be used to calculate the standard discriminability statistic between old and new items: d' = (mean of old match − mean of new match)/*SD* of new match.

A Modification to the Prototypical Connectionist Model: Serial Presentation

A major focus of the research reported here involves a departure from the usual training method for stimuli in a multilayer connectionist system. In the usual system, the vectors to be learned are trained in sweeps through the set of all the vectors to be learned. Thus, all vectors are available to the network throughout training. For example, if four vectors, **A**, **B**, **C**, and **D**, were to be learned, the system would be presented with the sequence **ABCDABCDABCDABCD** · · · until some learning criterion for the joint solution was reached. I define this as *remote rehearsal* because after item **D** is presented, the remote item **A** is brought back for another presentation. In domains such as perception, one can imagine repeated exposures to a set of stimuli to be learned that would approximate this remote sequence (in fact, some recent models such as Seidenberg & McClelland's, 1989, rotate training through stimuli according to their frequency of occurrence in English). In contrast, in recognition memory, when the items in a study list are presented at fast presentation rates, items are learned largely in the order in which they are presented, with little opportunity for remote rehearsal. For example, it is unlikely that a subject could retain all the items in a 16- or 32-item list in order to sweep through the whole list. Note that for remote rehearsal, the memory system that retained the whole list would need to be modeled within a connectionist framework, and this would lead to redundancy in memory systems (one to retain it for the other to learn it). In addition, it is possible to use study materials of a kind that cannot be easily rehearsed throughout presentation of the study list (e.g., pictures presented at fast rates). Because of these considerations, models in the class I examine use only local rehearsals; that is, only the item presented last is rehearsed, or only the last small group of presented items is rehearsed, where the number of items in the group is typically set to 4 (e.g., see the rehearsal scheme presented in Gillund & Shiffrin, 1984).

A second departure from many connectionist models is that only positive instances are presented to the system. Only old items that require a positive match from the system are presented at study; new items (requiring a negative response) are not presented for training. This means that the system receives no discriminative training between old and new items, only discriminative training among positive instances. Yet, the primary requirement for the system is to discriminate new items from old items.

The first set of studies I discuss explores forgetting in systems with small vectors followed by large vectors, and the second set of studies investigates amount of learning as a function of the number of learning trials.

Forgetting Resulting From Subsequent Learning: Interference Effects

Studies With Four-Element (Small) Vectors

The first set of simulations uses the encoder model with small vectors to examine the feature-by-feature behavior of the system before and after forgetting. There are four orthogonal vectors to be learned (1 0 0 0, 0 1 0 0, 0 0 1 0, and 0 0 0 1) and a multilayer system with four input units, four output units, and three hidden units (cf. Ackley et al., 1985). When all four vectors are learned as an ensemble (remote rehearsal) by repeatedly sweeping through the whole set, the system successfully learns them all. The learning criterion typically used is to continue sweeping until the output values are close to the input values, that is, within some small percentage of the input values, for example, 6%.

Interference is examined by observing what happens when three of the vectors are learned to criterion in sweeps (e.g., 1 2 3 1 2 3 1 2 3 1 2 3 · · ·), taking approximately 90 sweeps, and then the fourth (e.g., 4 4 4 4 4 · · ·) is learned to criterion by itself (approximately 40 presentations). The questions to be answered are whether and how performance on the first three vectors is degraded as a result of learning the fourth item. The results show high degrees of interference. These results are described in the next section, and then a sequence of modifications of the model follows, with each modification an attempt to reduce the interference.

Modifying all weights. In the first implementation, all weights were allowed to change during training of the first three vectors, and all weights were also allowed to change during training of the fourth vector. Table 1 shows the results of testing

Table 1
Modifying All Weights in the System

Training set	Test vectors	Response
1 0 0 0	1 0 0 0	.94 .04 .05 .02
0 1 0 0	0 1 0 0	.05 .94 .04 .02
0 0 1 0	0 0 1 0	.04 .05 .94 .02
0 0 0 1	1 0 0 0	.77 .01 .01 .90
	0 1 0 0	.01 .75 .01 .92
	0 0 1 0	.01 .01 .73 .92
	0 0 0 1	.03 .04 .04 .95
	1 1 1 1	.02 .03 .04 .95
	0 1 1 0	.00 .36 .33 .91

Table 2
Modifying Only the Small Weights

Training set	Test vectors	Response
1 0 0 0	1 0 0 0	.94 .04 .05 .02
0 1 0 0	0 1 0 0	.05 .94 .04 .02
0 0 1 0	0 0 1 0	.04 .05 .94 .02
0 0 0 1	1 0 0 0	.60 .05 .00 .63
	0 1 0 0	.01 .85 .01 .36
	0 0 1 0	.01 .01 .19 .98
	0 0 0 1	.04 .02 .05 .94
	1 1 1 1	.02 .02 .02 .94
	0 1 1 0	.01 .45 .11 .81

each of the four vectors after the fourth was learned and shows that there were serious distortions in the reproductions of the first three vectors. For each of these three output vectors, there was an entry for the feature trained for the fourth vector. So, for example, the response to 1 0 0 0 after training on the fourth vector (0 0 0 1) was 0.77, 0.01, 0.01, and 0.90. These results indicate that weights in the system were altered by the training of the fourth vector so that almost any test vector would evoke a response in the fourth feature position. Thus, any output is a blend of the correct output and the vector that was trained last; an item presented earlier for test retrieves the last-studied item modulated by the earlier item. The general conclusion is that training new vectors after training an earlier set, with no repetitions of the earlier set, will result in loss of the ability of the network to reproduce the earlier set. (Also included in Table 1 are the results for testing two other vectors, 1 1 1 1 and 0 1 1 0. These are included to show what happens when a new item is tested; in this example, the last feature is turned on for these new items as well as for the old items.)

Modify only low weights. After the items in a set have been learned to criterion, the weights on the connections between units will vary in how much they have been modified from the initial values. For the system to learn more items, perhaps the weights that were modified least in original learning should be used for new learning, leaving the other weights unaltered (the notion would be that altered weights become more resistant to change, or less plastic). To implement this scheme, three items were learned to criterion (as in the previous simulation). I then empirically determined a cutoff value of weight strength so that approximately half the weights were large (either positively or negatively); these were not modified during training of the fourth item. The other half of the weights were small and could be modified. Table 2 shows results parallel to those in Table 1 for the output vectors produced when each of six input vectors are tested. The results show somewhat better reproduction of the input vectors than when all weights are modified (cf. Table 1), but the improvement is not large and does not change the general conclusion that later learning distorts earlier learning. Note that reproductions of the diagonal elements for the first three patterns after training the fourth vector were quite variable (0.85–0.19). This is a joint effect of dividing the weights into two classes, fixed and adjustable, and of the variability in starting values of weights. If a different set of randomly selected initial weights are chosen, the diagonal elements are just as varied, but instead of the second vector reproducing itself well, the first or third vector might.

Add new hidden units. If neural systems are taken seriously as the inspiration for connectionist models, then the notion that uncommitted nodes and connections can become committed during learning is worth examination. Bringing in uncommitted nodes is simulated by assuming that even after considerable training, there are still uncommitted nodes available to be recruited during further training. By this model, learning new information is viewed not as reorganizing already existing memory but as training new resources alone or as modifying old memory plus new resources. These two possibilities are examined in the next simulations.

The first simulation adds one extra hidden unit for training the fourth vector, and all the weights in the network are allowed to change. One might expect that much of the new learning would be accomplished through the new connections. However, although the results (Table 3) show that recovery of the first three vectors is better than in the simulations described earlier, performance is not perfect. For example, after learning 0 0 0 1, responses in the fourth position of the first three vectors (originally 0) varied from 0.46 to 0.50, indicating considerable interference.

Add new hidden units and modify only weights on connections to and from the new units. In the simulation just described, the network as a whole (the old connections plus the new connec-

Table 3
Adding Extra Hidden Units and Modifying All Connections

Training set	Test vectors	Response
1 0 0 0	1 0 0 0	.94 .04 .05 .02
0 1 0 0	0 1 0 0	.05 .94 .04 .02
0 0 1 0	0 0 1 0	.04 .05 .94 .02
0 0 0 1	1 0 0 0	.86 .01 .01 .46
	0 1 0 0	.02 .85 .01 .50
	0 0 1 0	.02 .02 .82 .49
	0 0 0 1	.04 .04 .04 .94
	1 1 1 1	.03 .04 .05 .94
	0 1 1 0	.00 .48 .41 .59

Table 4
Adding Hidden Units and Modifying Only the Connections to the Hidden Units

Training set	Test vectors	Response
1 0 0 0	1 0 0 0	.94 .04 .05 .02
0 1 0 0	0 1 0 0	.05 .94 .04 .02
0 0 1 0	0 0 1 0	.04 .05 .94 .02
0 0 0 1	1 0 0 0	.93 .00 .01 .13
	0 1 0 0	.04 .60 .01 .15
	0 0 1 0	.04 .00 .80 .15
	0 0 0 1	.17 .00 .03 .51
	1 1 1 1	.12 .00 .04 .49
	0 1 1 0	.01 .19 .38 .20

tions to and from the new hidden unit) could not preserve learning of the first three items when the fourth was trained. In the next simulation, only the weights to and from the new hidden unit were modified during training the fourth item. At final test, all connections entered into calculating the output values for a test item. Results are shown in Table 4, and performance is similar to that shown in Table 3. The main difference is that the fourth item did not reproduce itself as well at final test because training did not include the whole network. However, the three initial vectors reproduced themselves better than in the earlier simulations. These two results suggest a tradeoff: The last vector can be reproduced better at the expense of earlier vectors or the earlier vectors can be reproduced better at the expense of the last vector.

More than one interfering item. In the simulations described so far, three items were trained to criterion first, and then a fourth was trained to criterion. In the next simulation, fewer items were trained initially, and more items were trained subsequently to determine whether this would result in poorer reproduction of the initially trained items. First, the vectors 1 0 0 0 and 0 1 0 0 were trained to criterion, and then two more vectors, 0 0 1 0 and 0 0 0 1, were trained to criterion. The results for the test items are shown in Table 5. The disruption to reproduction of the initial two items after training the last two is large. In fact, comparing the response to the two new vectors tested (1 1 1 1 and 0 1 1 0), there is little information in the output to identify the initially trained vectors. The conclusion from this simulation is that the greater the number of vectors trained in a group after original training, the less likely the earlier trained vectors are to be accurately reproduced.

Summary for Four-Element Vectors

Each of the simulations showed that when items or sets of items are trained sequentially, training on the later items disrupts learning of the earlier items. The simulations used small vectors of length 4, with one set of items trained to criterion and then a second set trained to criterion. Results showed that a test of an item from the first set after training the second set resulted in massive interference: The output vector was more similar to one of the items from the second set than it was to the test item. Attempts to reduce interference by modifying only some weights, adding hidden units, and coding the two training sets by different hidden units reduced interference a little but did not solve the interference problem.

McCloskey and Cohen (1989) have obtained similar results for two different tasks: learning arithmetic facts (using a coarse-coding vector representation) and the Barnes and Underwood (1959) A–B, A–C interference paradigm. McCloskey and Cohen conceptualized the rapid forgetting caused by interference in terms of a multidimensional space (an energy or sum-of-squares space), where each dimension in the space is a connection weight and where the learning algorithm searches through this space for a minimum energy. The learning algorithm adjusts the weights to bring the sum of the differences between the values of each output feature and its training signal to a minimum (e.g., an energy minimum). Thus, learning a set of items involves gradient descent in a multidimensional space where each dimension or weight is gradually adjusted to produce the energy minimum. If the number of hidden units is large, then there will be several (if not very many, e.g., 10^9) solutions; that is, there will not be a unique minimum (Pavel & Moore, 1988). Then, the particular solution obtained for one set of items will serve as the starting point for training of subsequent items (see Grossberg, 1987), and the new solution will be near the old solution in the multidimensional space. This new solution will not generally be a solution that maximally satisfies both the set of constraints on the initially learned items and the constraints for the subsequently trained items; rather, it will satisfy only the last items trained. Thus, reproduction of the earlier items will be degraded. The initially trained items will be reproduced to some extent because the solution to the new set of items is not too far from the solution for the earlier learning. In contrast, when a set of items is to be learned all at once, the method of sweeps leads to an iterative path through the parameter space to converge on a joint solution.

In the simulations I have described, attempts to constrain the space of possible solutions by allowing only a subset of the weights to be modified were of limited success. The reason for this is that the network is highly interconnected, and learning is distributed across the whole network so that all weights, both small and large, are carrying part of the solution. The same conditions hold in the simulations in which extra hidden units and connections are added. The solution is still carried by the whole network, and despite adding new resources, there is still movement of the whole system through the solution space to a new

Table 5
Training Two Vectors, Then Two More, Modifying All Weights

Training set	Test vectors	Response
1 0 0 0	1 0 0 0	.95 .04 .02 .02
0 1 0 0	0 1 0 0	.06 .94 .05 .05
0 0 1 0	1 0 0 0	.05 .00 .90 .15
0 0 0 1	0 1 0 0	.00 .40 .08 .87
	0 0 1 0	.05 .01 .94 .06
	0 0 0 1	.00 .03 .05 .95
	1 1 1 1	.00 .03 .05 .95
	0 1 1 0	.00 .10 .57 .30

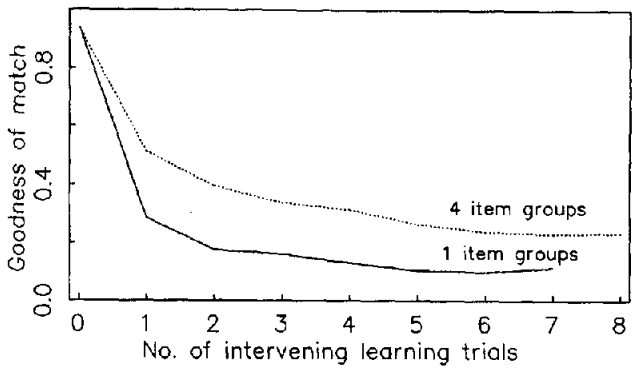

Figure 2. Results for a simulation examining forgetting functions with larger vectors in which all weights are modified in training.

point. The only difference between these latter simulations and the earlier simulations is that reproduction of old vectors is somewhat better because, with extra resources, a solution can be found nearer to the starting point (the prior solution).

Forgetting and Interference in a Larger System

The simulations with a small set of short, orthogonal input vectors allowed the effects of new learning on prior learning to be observed easily and conceptualized feature by feature within a vector. However, a limitation of these simulations might be that such short vectors do not provide a complex or rich enough system to acceptably mimic the human processing system. Most vector models of memory use vectors of length greater than 10 (because long vectors would be needed to represent all the words in a mental dictionary, for example) and assume that properties of the solutions scale up; that is, they remain qualitatively the same as vector length is increased. For the next set of studies of forgetting, I used a system with input and output vectors of 32 units and a hidden layer of 16 units. Only a maximum of 12 items were trained so that rapid forgetting could not be attributed to lack of resources in the network (i.e., a vector length too short for the number of items to be learned). In addition, the vectors for the items to be trained were not orthogonal; values for each feature for each item were selected randomly so that random correlations between items would be present as they would be with real items in memory.

The simulations presented in this section parallel the manipulations examined in the first section with simulations in which all weights are modified, subsets of weights are modified, or extra hidden units and weights are added to the system for each new item to be learned. In these simulations, the system was first presented with four vectors in a group, with the elements of each vector chosen randomly to be either 0 or 1. Training sweeps through the whole group continued until the vectors were learned to criterion. Then, eight random vectors were trained individually. After each of the items reached criterion, that item and all previously trained items were presented for test. The match between each of the trained items and its output was computed, yielding forgetting functions for the four initially trained items as well as the successively presented single items. The items were learned to a criterion of 0.94 (unless otherwise noted), and a typical range of training trials was 50–120. Twenty replications of this basic simulation were performed to generate estimates of variability in performance; the results reported are based on means over these replications.

For the first set of simulations, all connections were available for modification during each training trial (both for the initial four items and for each successive new item). The results are presented in Figure 2 and show that there was rapid forgetting for the initially learned four items as a function of successive learning trials on the single items. The match value fell from 0.94 to 0.51 after training one intervening item. By six or seven intervening items, the match had fallen to 0.23. For singly presented items, after one intervening item, the drop in match was worse, falling from the initial value of 0.94 to 0.29, and after six or seven further items had been learned, the match dropped to 0.10. This last value is close to the match for new items (random vectors with the same statistical properties as the trained vectors). The match for new items was 0.125 when they were tested immediately after the four initially trained items and 0.046 ± 0.046 when they were tested after the successive single items.

The decrease in the match value for new items can be understood by considering how well the new items match the trained items. For a new item tested immediately after the initial four, its output will be like the item of the first four that is most similar to it: The new item is trying to reproduce the item most like it. The output will match the test item to the extent that both are similar. For new items tested after single items are trained, the last item trained (or with lower probability, the item before) will be reproduced to some degree, and this will match the last item learned best (or to a smaller extent, the item before). This will lower the average match relative to testing after the initial four are learned because by chance, one of the four will match better than only the last one (or two).

Overall, from this set of results, two conclusions can be drawn: First, as with the short four-element vectors, items learned singly are poorly reproduced after several intervening learning trials. Second, items encoded in groups are significantly more resistant to forgetting than items learned individually.

In the second set of simulations, the items learned individually (after the initial four items) were trained with only the low-valued weights altered. To accomplish this, the weights were divided in half after examination of the results of training the four initial vectors. The smaller weights were allowed to change during training, but the larger weights were not. The cutoff value dividing the large and small weights was fixed for subsequent learning of individual items so that when any weight exceeded this value, it was not allowed to change during training of any further items. Occasionally, the system would fail to reach the learning criterion for one of the later individual items. In this case, 1,000 training sweeps of the item were completed to ensure that training asymptoted before proceeding to the next item. Results presented in Figure 3 show that the asymptote is higher than the corresponding asymptote in Figure 2, both for items learned in the group of four and items learned singly. For new random vectors tested immediately after the first four, the mean match was 0.125, and for new items tested later, the mean match was 0.045 ± 0.039. Although performance was

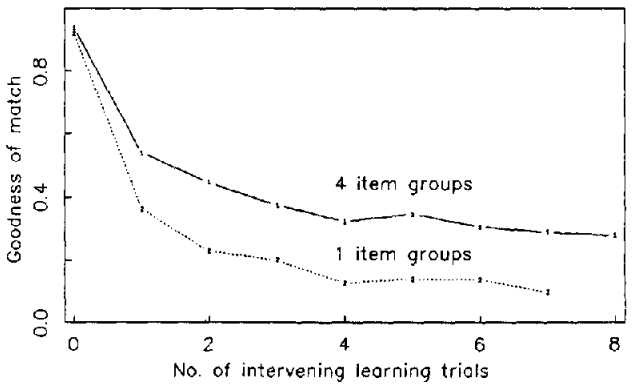

Figure 3. Results for a simulation examining forgetting functions with larger vectors in which small weights are modified in training.

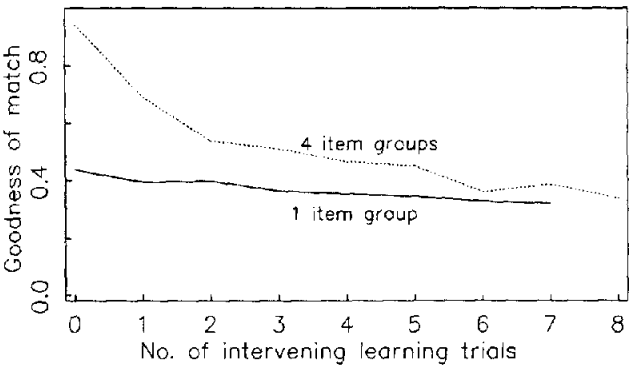

Figure 5. Results for a simulation examining forgetting functions with larger vectors in which two hidden units are added per new item learned and weights are updated in isolation from the rest of the network. (At test, all units and weights participate.)

better than in the first study, there was still rapid forgetting of highly learned information as a function of later learning.

The third study was designed to examine the issue of adding resources to the system: Two new hidden units and their connections were added for each additional item to be learned. Each additional item was trained by modifying only the weights to and from the new hidden units, but connections to and from all the hidden units determined the output during training. Items in the initial group were retained better than in the first set of simulations (Figure 4) because the weights to and from the 16 hidden units were not altered in later training and they dominated over the effects of new learning. The single items learned subsequently were retained much more poorly for the same reason. By the time seven or eight subsequent items had been trained, the match for these items was close to 0.

In the fourth study, two new hidden units were added for each single item to be learned. Only connections to and from these two new hidden units were altered during training, and only the new hidden units were used to determine the output during training, so that training of the single items was carried out independently of the rest of the network (this contrasts with the previous simulation, in which connections to and from both the old and new hidden units were used to produce the output during training). At test, all connections were included to produce outputs. As in the example in Table 3 for the four-element vectors, immediate tests of the single items learned last did not give high matches because previously set weights were involved at test, but these weights had not been involved in training the item. Also, as in the two studies discussed earlier, there was a rapid drop in the match for the initial four items as a function of subsequent learning (see Figure 5). However, compared with the first study, the match was better for both the initial items and the single items because the effects of interference were not in learning but at test, where subnetworks (the different hidden units and weights used in learning the different items) were combined.

To show that this result is due to keeping learning separate and combining only at retrieval and not due to simply providing more hidden units, a further simulation was carried out with all weights modified and more hidden units provided (32 instead of 16). Figure 6 shows the results. Values of match for items encoded in the 4-item set were only a little below those in Figure 5 when the new hidden units were trained in isolation, but the single-item match was approximately half that of the values shown in Figure 5. In fact, performance is comparable to that in Figure 2, showing that extra resources have little effect on performance.

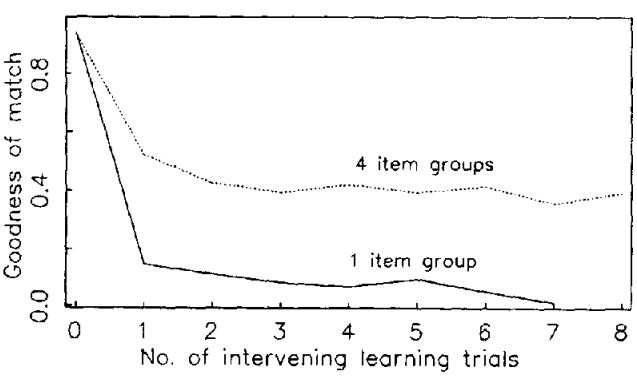

Figure 4. Results for a simulation examining forgetting functions with larger vectors in which two new hidden units are added per learning trial and all units provide input for weight modification and match values.

Summary of Interference in a Larger System

Results showed that if a set of items is trained and then another item is trained without retraining items from the first set, then there is a huge drop in recognition performance on the first set. In addition, recognition performance for a single item, trained to criterion after the first set, drops even more after another single item is trained to criterion. There was little improvement when low weights were trained while keeping larger weights fixed. There was a reduction in the amount of interference when extra hidden units were added to the system for each item trained, but only when the rest of the network was held fixed while each new item was trained. When all weights were allowed to change, increasing resources (numbers of hidden units and connections) did not improve performance.

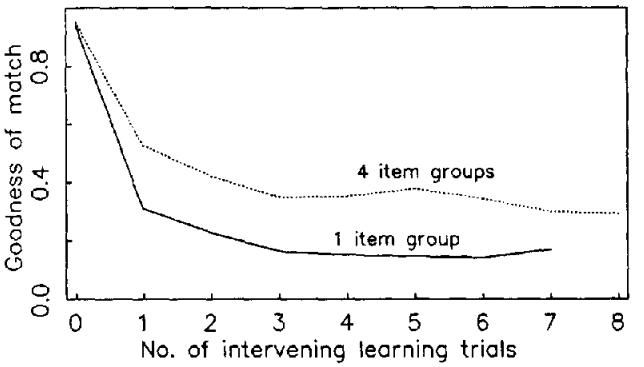

Figure 6. Results for a simulation examining forgetting functions with larger vectors and twice the number of hidden units (32) in which all weights are modified in training.

Discussion of Forgetting Functions

The rapid decay of early learning as a function of interference from later learning suggests that the encoder model is inadequate as an account of recognition memory. The model produces forgetting of well-learned information at a much faster rate than that found for human memory. After learning three or four items to a high criterion, learning only one new item to criterion causes the earlier items to be reproduced poorly. This is like studying the word *cat* 100 times, studying the word *table* 100 times, and then finding that *cat* is not well recognized and that when *cat* is presented for recognition, *table* is retrieved (see Grossberg, 1987).

Although forgetting in the above simulations is clearly too rapid to model forgetting in human memory, the discriminability between old and new items after the initial rapid forgetting is examined in the next sections to provide information about another basic characteristic of human memory. In the multilayer model, after several interfering learning trials, d' values from the simulations with 32 feature vectors are in the range of 1 to 3, and these values are consistent with human performance in list-learning memory experiments. So, despite the serious problems with rapid forgetting, other aspects of performance of the model are still worth examining to gain further understanding of its properties and structure in case some solution to the forgetting problem can be discovered. The issue of amount of rehearsal is considered in the next section.

Learning as a Function of Amount of Rehearsal

The second main set of studies examines old–new discrimination as a function of the amount of learning (defined as the number of learning trials). Results from experimental manipulation of amount of learning have been important in memory research because they have provided a diagnostic for classes of models. Linear-distributed (vector) models such as those of J. A. Anderson (1973) and Murdock (1982) have a serious problem in that old–new discrimination does not increase as a function of amount of learning (although see Murdock, 1989). In these models, if each item in a list is presented once, then the d' value will be the same as if each item were presented several times. The reason for this is that even though the mean of the match for old items increases as the number of presentations increases, the standard deviation in the match for new items also increases to keep d' constant (i.e., means and standard deviations are all scaled up equally as a function of the amount of learning). Because constant old–new discriminability is a key problem for some memory models, predictions from the multilayer encoder model are evaluated with this diagnostic.

Theoretically, in the encoder model, the behavior of old–new discrimination as a function of amount of learning is related to the problem of rapid forgetting demonstrated in the last section. When items are learned sequentially, memory for items learned earlier is substantially degraded. The information left in the system is the modulation that remains from the learning of earlier items superimposed on the learning of the later items. The implications of this forgetting problem for old–new discrimination are examined in the following simulation studies. Specification of the encoder model requires implementation of a rehearsal scheme, and that is described first. Then, the next section presents an overview of the results obtained from the model, followed by detailed discussions of the results. Finally, possible modifications of the basic model are considered.

Rehearsal Buffer Model

Empirical work has shown that in list learning, groups of items are rehearsed together (Atkinson & Shiffrin, 1968; Rundus & Atkinson, 1970). To provide group rehearsal for the encoder model, a simplified rehearsal buffer was used (as in Gillund & Shiffrin, 1984). The items of a list to be learned are entered into the buffer in small groups (e.g., four items). Then, each item in the buffer is rehearsed for some number of sweeps (N = the number of learning trials) across the whole buffer. After N sweeps, the first item in the group is dropped out, and the next item in the list is added to the buffer, and another set of learning trials proceeds (another N sweeps across the buffer). This process continues through the list of items. The studies presented earlier show that items learned in groups are more resistant to forgetting than items learned singly, so one advantage of training items in groups is that memory performance is improved. Of course, the rehearsal process is also designed to fit with existing knowledge about human rehearsal processes.

Specification of the Model

The multilayer encoder model to be used in this set of studies has 32 input and 32 output elements and 16 hidden units, input vectors have randomly selected elements that are either 0 or 1, and buffer size is either 1 or 4 items. When the buffer size is 1 item, each item is trained N times before moving to the next item to be trained. When the buffer size is 4, the first item to be learned is encoded by itself on each of N sweeps. Then, the second item is presented and encoded with the first item on each of N sweeps, and then the third item is presented, and all of the first 3 items are encoded. After that, each of the N sweeps through the buffer for each group has the current item plus the 3 prior items. Thus for N sweeps, each item is presented (buffer size) $\times$ N times. Thus, if number of weight updates for different buffer sizes are to be compared, the number of sweeps must be

scaled by buffer size. The total list length is 16 items. The first 12 items are used in measuring performance, and the last 4 items are used to eliminate short-term memory effects. All testing of the system occurs after all 16 items have been trained. At test, the first 12 items are presented, and these provide a mean and standard deviation in the match for old items. Twelve randomly selected new items are also tested, and these provide the mean and standard deviation for new items. In most of the following simulations, the results shown are for the averages of 100 replications of each condition.

Overview and Summary

Several factors need to be examined to provide explanations for the results presented later. This section summarizes the main points and provides a framework for the results. In essence, an understanding of the model's performance begins with the finding from the previous section that studied items are quickly forgotten when other items are trained. When the system is given a test item from earlier in the study list, it does not produce anything similar to that item as output. Given this finding, then, the questions to be addressed are, What does the system produce in response to a test item? and How does this output vary with amount of learning?

To begin the overview, the simplest result is for Buffer Size 1: The system rapidly forgets each item after training, and any test vector (old or new) reproduces the last vector learned to a greater extent as amount of learning increases.

When the buffer size is 4, so that items are rehearsed in groups, the results are more complicated and depend on the learning rate, that is, on the amount the weights are altered each time an item is trained (the learning rate η is a multiplicative constant in Equation 4 and controls the amount the weights are modified). For low learning rates and few learning trials, the system learns to reproduce an average or prototype of the vectors that are most similar to each other in the last group of four rehearsed. Both old and new test items produce this prototype (to a greater or lesser magnitude) when they are tested. With larger numbers of learning trials, higher learning rates, or both, old and new test items reproduce one of the last four items rehearsed (not the prototype of the four), the item to which they are most similar. It is as though the system has set up four attractor states, and any test vector is drawn to the state most similar to itself in producing an output. This transition from a test item reproducing a prototype of the most similar of the last-learned vectors to a test item reproducing a specific vector corresponds to the transition from largely linear behavior of the net input-to-activation transformation (Equation 2), when net inputs are small, to significant nonlinear behavior, when some net inputs are very large, either positive or negative. Significant nonlinear behavior is the result of large values of weights either from many learning trials at lower learning rates or from more than one or two learning trials at high learning rates.

Although an old test vector from early in the list best reproduces either one of the last items studied or the prototype of the last items studied, this output is modulated by a systematic deviation that shows a residual effect of the test vector itself. Figure 7 shows this effect for a list of two items. The top panel shows the vector studied first in the list, a sawtooth, with lines

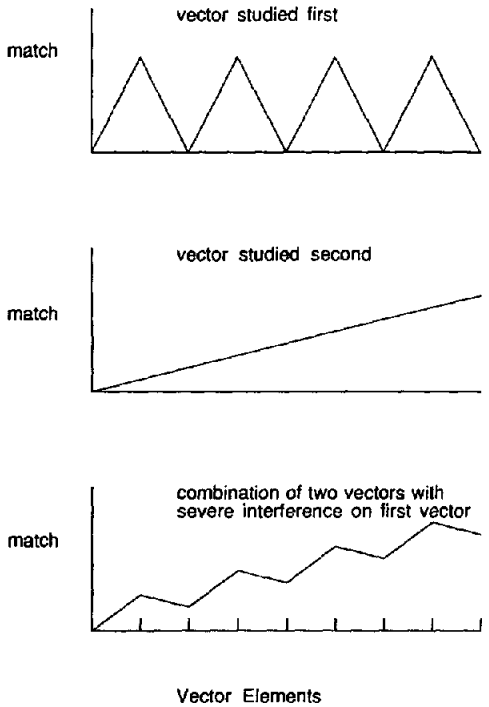

Figure 7. An example in which one vector is learned; a second vector is learned, interfering with the first vector; and the output is a combination of the two (in the models examined here, this would represent the result of testing the first vector).

joining the adjacent elements to show patterns; the middle panel shows the vector studied second (and last), a ramp; and the bottom panel shows the output reproduced in response to the sawtooth test item with a reduced (through interference) sawtooth superimposed upon the ramp. This figure illustrates the reproduction of the vector learned last dominating at output, with memory for earlier items as a modulation on that output. This behavior shows that the system could not perform a recall task (only the item learned last would ever be recalled). However, the system could still have potential validity for recognition because recognition requires a different behavior, namely discrimination between studied old items and nonstudied new items.

For new test items, an unexpected result is that the value of match between a new test item and memory increases as a function of the number of learning trials for the studied items. The system seems to be learning to reproduce the new test item better as a function of training on study items. However, this is not the correct interpretation. At high learning rates, the value of match for a specific new item depends on the extent to which the new item is similar to an old item in the last rehearsal group. Over all the items in the last rehearsal group, the average match between the new item and an item in the group is 0. However, of these match values, some will be greater than 0 (by chance), and one of these will be largest. This item is the one that is produced by the system in response to the new item. The more learning trials the old item (and its rehearsal group) has re-

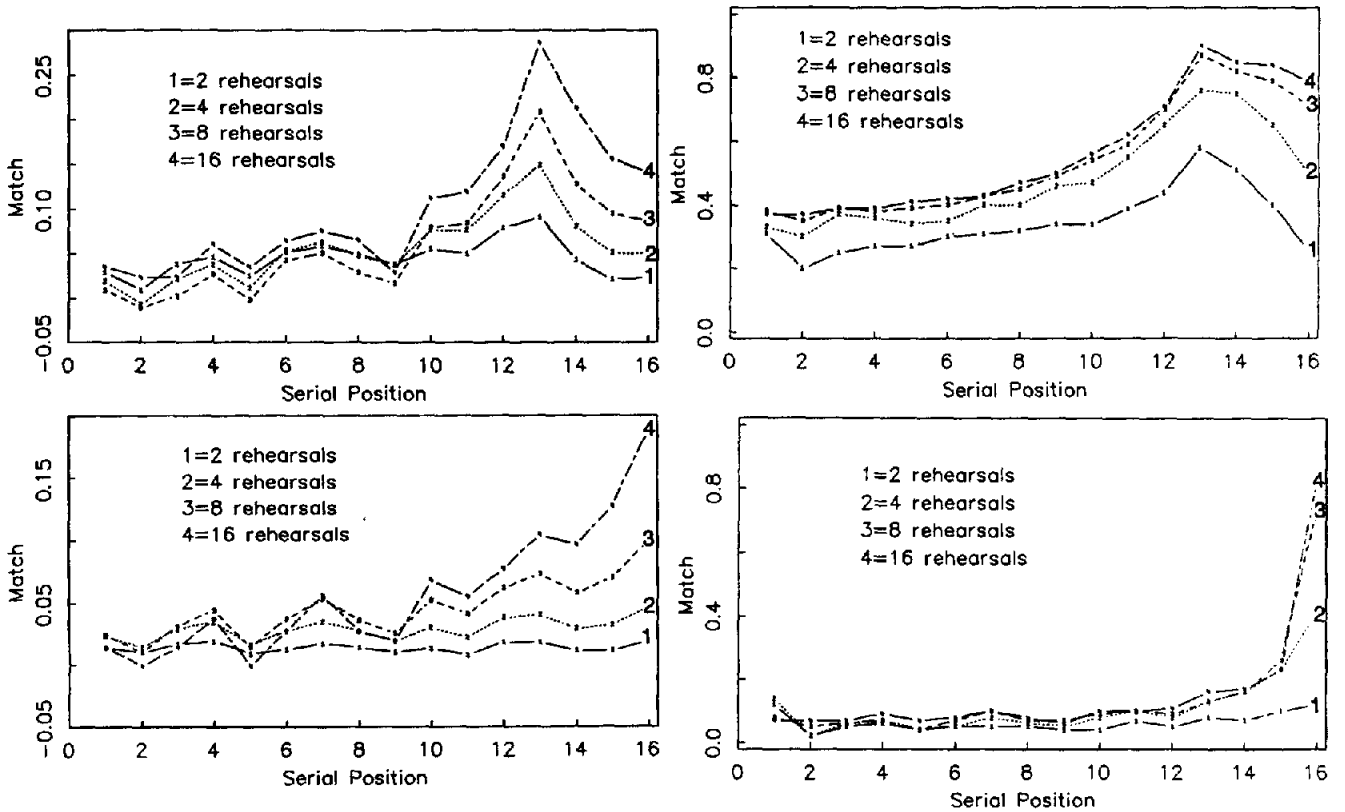

Figure 8. Serial-position effects for learning rates of 4.0 (weight starting values ± 0.3) and 0.05 (weight starting values ± 0.05; right column and left column, respectively), with one- and four-item rehearsal groups (bottom row and top row, respectively).

ceived, the better it will be produced in response to the new item, and the greater the match.

At low learning rates with few learning trials, new test items give the same behavior as old test items; they reproduce the prototype of the last group of old items. Because only the one prototype vector is retrieved, the average value of match of new items will be 0 because a new item will match the single-prototype 0 on average.

These patterns of behavior for old and new test items are complicated and were not predicted before the results were observed. These patterns are demonstrated in the detailed presentation of the results below. The reader not needing details of the summary just presented can skip to the Effects of Similarity Among Vectors section.

Serial-Position Effects

The first set of results to examine are serial-position effects for old test items. These provide the primary evidence for the dominance of the last learned items. Figure 8 shows the values of match for old test items as a function of serial position for Buffer Sizes 1 and 4 and two learning rates (high = 4.0; low = 0.05). The patterns of results are similar across learning rates with a 1-item recency effect for Buffer Size 1 and a 4- to 6-item recency effect for Buffer Size 4. For Buffer Size 4, the value of match as a function of serial position peaked at Item 13 (because it is the item that was rehearsed most often in the last and earlier rehearsal groups) and fell from there to the last item. The serial-position functions in Figure 8 show the same profiles (e.g., Item 4 match greater than Item 3 match) across buffer size, number of learning trials, and learning rate because the same randomly selected set of vectors was used in each simulation, leading to consistent variations across serial position. The use of momentum in the learning algorithm had small effects on the serial-position functions. It led to slightly elevated match values on earlier items in the recency part of the curve and slightly reduced values of match for the last item. Thus, setting momentum to 0 tended to make the recency effect a little shorter.

Old–New Discriminability as a Function of Learning

The next results to be reported are d' values as a function of amount of learning. Understanding the d' values requires that the results be broken down into values of match for old and new test items, and these values are also presented. Figure 9 summarizes the results for d' as a function of the two variables, learning rate and number of learning trials, for Buffer Size 1. For high or medium learning rates, d' decreased or was constant as a function of number of learning trials. For low learning rates, the d' function was nonmonotonic: As the number of learning trials varied from small to large, d' first increased to a

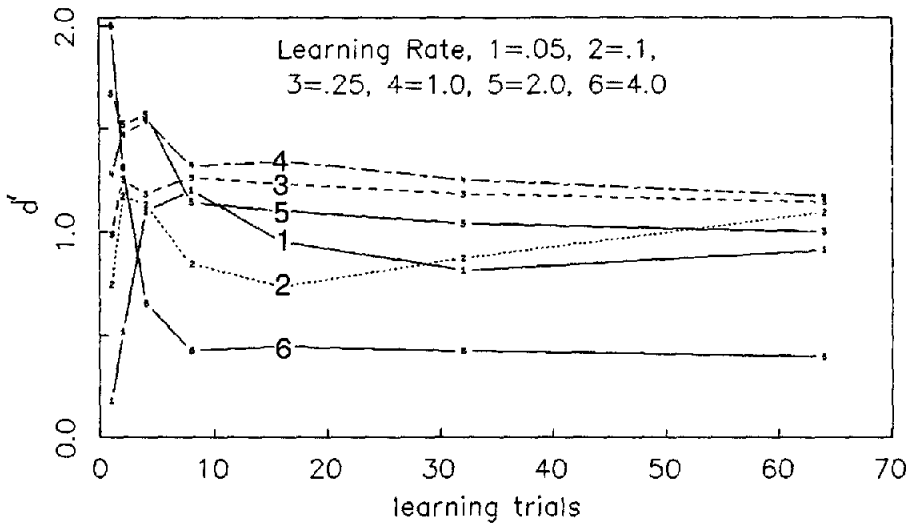

Figure 9. Old–new discrimination as a function of number of learning trials and learning rate for Size 1 rehearsal groups.

peak, then decreased, and then increased (to asymptote for 64 learning trials). The reason for this nonmonotonicity is discussed later.

For four-item rehearsal groups (in Figure 10), at low learning rates, d' was nonmonotonic as a function of the number of learning trials; as the number of learning trials increased, d' was either initially constant or decreased, and then it increased to an asymptote. For high learning rates, there was a gradual decline in d'. Figure 11 contrasts d' functions for which the initial starting weights were small versus large and the learning rate was low. When the values of the initial weights were small, d' increased to an initial peak by two learning trials, then decreased followed by a rise to asymptote.

An explanation of these d' effects requires examination of the components of d', the values of match for old and new items. The results were similar for the two buffer sizes. For a low learning rate and low starting values of the weights, the match value for old items increased, decreased (corresponding to the drop in d'), and then increased again as a function of the number of learning trials (see Figure 12). The match for new items stayed approximately at 0 until d' began to rise after the early fall in d', and then it increased. The standard deviation in the new-item match increased steadily. The precise form of the changes in these three quantities gave rise to both the monotonic and nonmonotonic behavior of d' as a function of the number of learning trials. For higher learning rates (e.g., $\eta = 2$ and $\eta = 4$), the values of the initial weights were less important because the weights were driven to extreme values with relatively few learn-

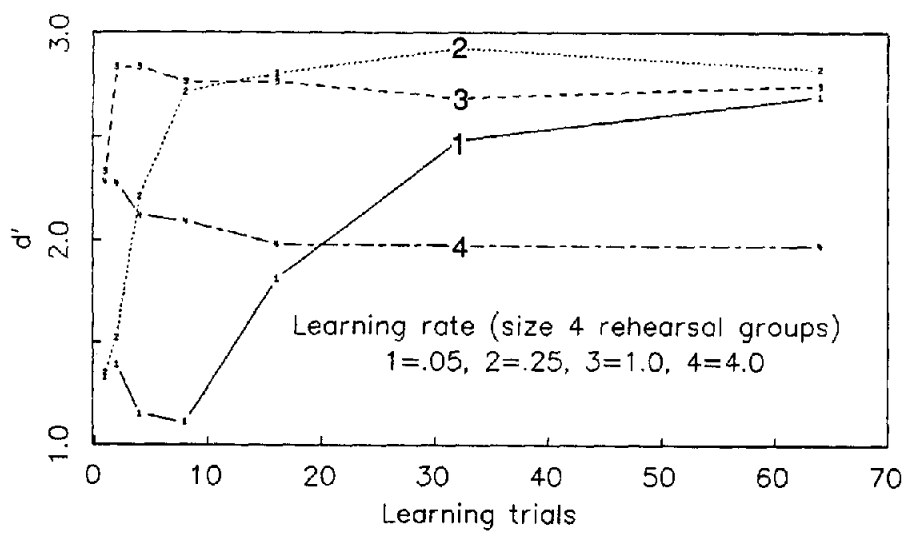

Figure 10. Old–new discrimination as a function of number of learning trials and learning rate for Size 4 rehearsal groups.

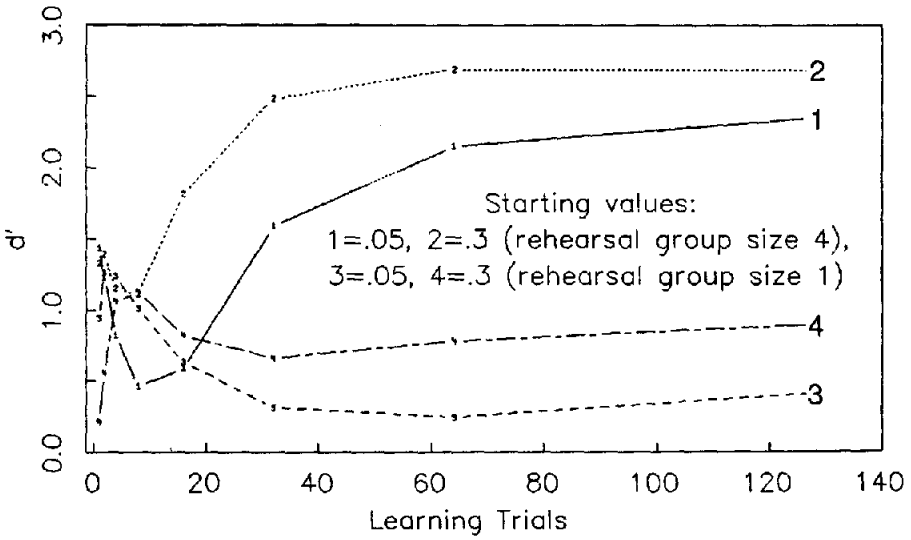

Figure 11. Old–new discrimination as a function of number of learning trials and starting values of initial weights. (Learning rate is set at 0.05.)

ing trials. An explanation of these patterns is given in the next section.

Another property of the system that is displayed at high learning rates concerns the lower asymptote in d' for larger numbers of learning trials. There are two sources that depend on the inability of the system to modify extreme weights. After large amounts of training, some weights become extreme. This results in small modifications to the weights because the error sig-

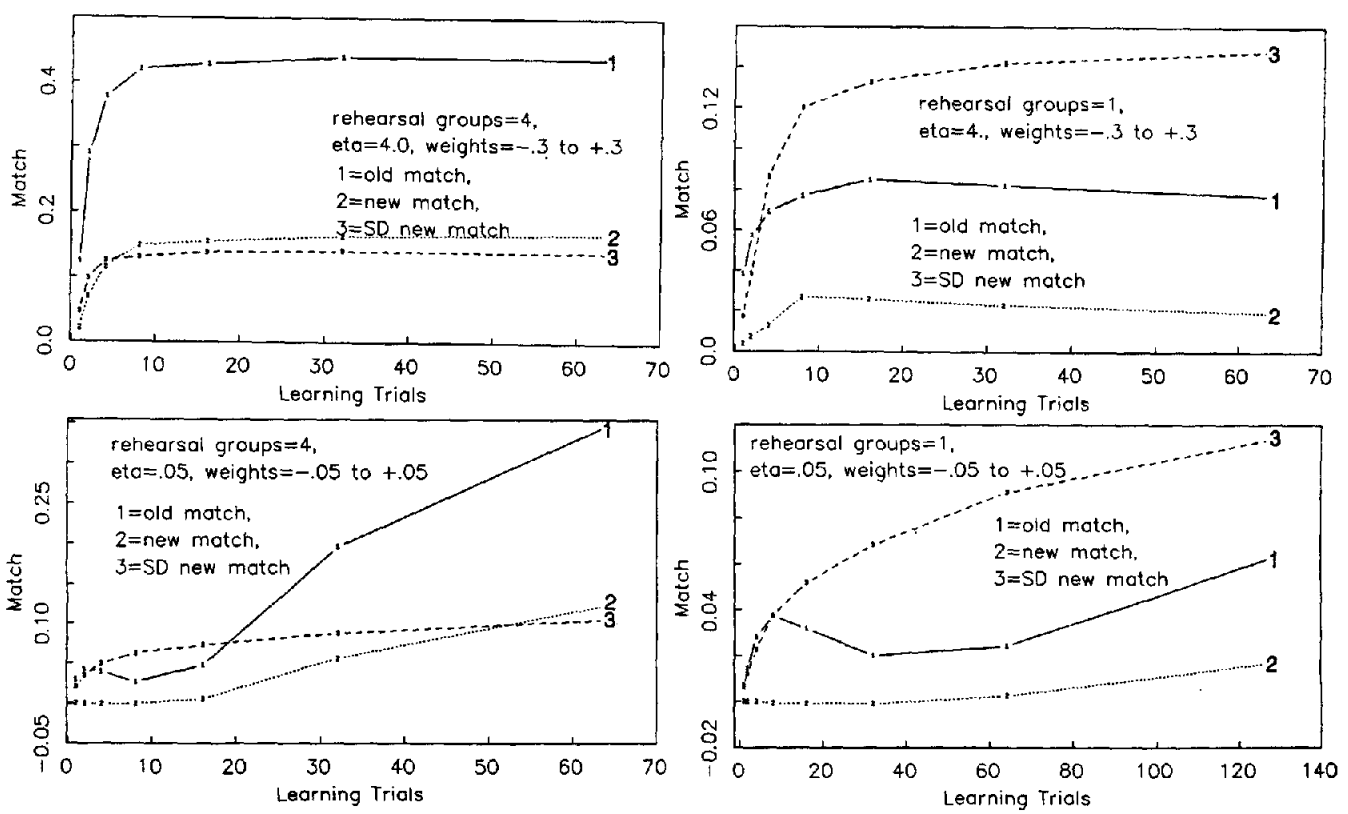

Figure 12. Match values for old items and new items, and standard deviation in new match as a function of learning trials.

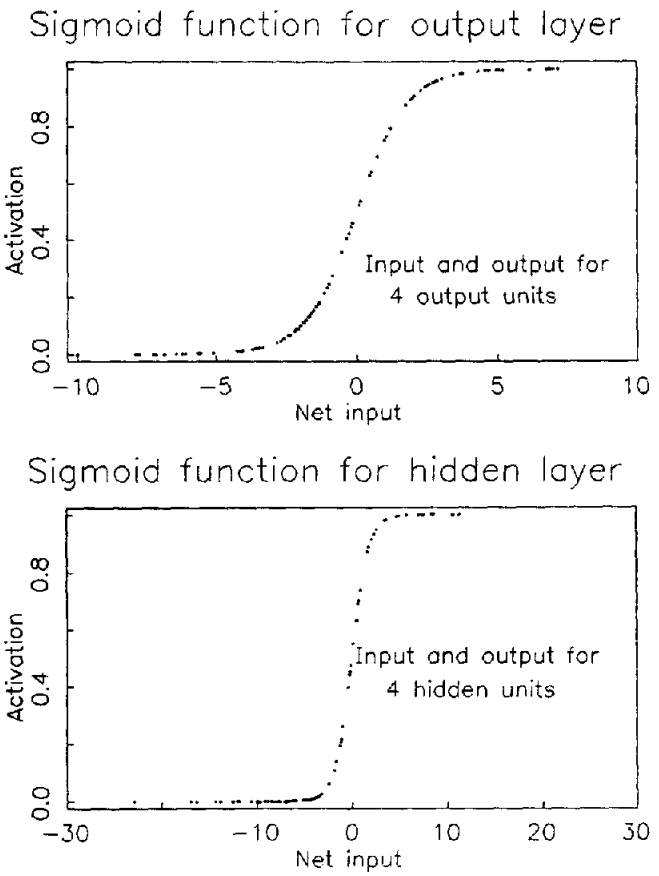

Figure 13. Values for the transformation from net input to activation for hidden units and output units for learning rate of 4.0, starting weights ± 0.3, and rehearsal groups of Size 4.

nal involves a term $o_i(1 - o_i)$, and when the weight has large magnitude, o_i, the activation value is either near 1 or near 0, leading to small changes in the weight (see Equation 3). As more weights become extreme, first, items to be learned that are inconsistent with the weights already driven extreme require many more learning trials to be learned (and so for fixed numbers of learning trials, performance is reduced). Second, there are fewer weights available for modification as learning proceeds. Both of these factors contribute to the fall in asymptote as learning rate increases (see Figure 9).

To demonstrate that high learning rates do give rise to extreme values of weights and thus activation values near 0 or 1, values of the net input and output (activation) for a typical simulation with four-item rehearsal groups and two learning trials per rehearsal group were used. Plots of the net input to a layer against the activation output are presented in Figure 13 (these correspond to input and output values of the sigmoid function shown in Equation 2). For the net input to and output from the hidden layer, there were approximately 22 of 96 units not at extreme values, and for the net input to and output from the output layer, there were approximately 47 of 96 units not at extreme values. These results show that the system has many units not easily able to be modified. In addition, for the model to mimic a linear model (such as discussed earlier with low learning rate and few learning trials), most points in the plots of Figure 13 would have to fall between the extremes, in the roughly linear region. (This also shows that the model is far from mimicking a linear model like those of Murdock, 1982, and J. A. Anderson, 1973).

Examination of data like those shown in Figure 13 (net input and the corresponding activation) shows that for low learning rates, and few learning trials, there are no values of activation that are extreme, and all points lie on the approximately linear part of the net input to activation function (e.g., between activation values of 0.2 and 0.8). When match values for old items drop and match values for new items rise above 0 (with increasing numbers of learning trials), some values of activation begin to reach ceiling and floor. This also corresponds to the transition from reproduction of the prototype to reproduction of one of the last few items learned.

Figures 9 through 12 show d' functions and match values for old and new test items. Now the question to be addressed is why the old and new functions show these particular forms. First, it is necessary to show what is produced by the system in response to a test item (old or new). The output of the system does not match the test item as well as it matches the item learned last (see Tables 6 and 7). For Buffer Size 1, the output matches the item learned last much better that it matches the test item itself. For Buffer Size 4, Table 6 shows the match of the output for the test item against the last studied item. The output matches this last-learned item better than it matches the test item itself. However, the last item is only one of the last group of four rehearsed, and a test item will reproduce the one of the last group that matches it best. This output will match even better than for the single last item.

For the higher of the learning rates, new items match the item learned last better than do the old items. This is because old test items produce an output that is a combination of the old item itself and the last-learned item, and this leads to a lower match to the last-learned item compared with new test items (see Figure 7).

When the output of the system for a test item matches a last-learned item better than it matches the test item, the reason could be that the output is matching a specific last-learned item or a prototype of the group last learned. To examine this, the results can be examined for just one replication for a new test item. For a low learning rate, the output for a new test item matches a prototype of the most similar to each other of the last four learned items. This is demonstrated in Table 8. The argument is as follows: If the response to new test items is a prototype, then each of the last four studied items will match the output given to the test items in the same ratio. If the response was not a prototype, then the output for each test item would match the last four studied items differently. (Note that the same argument holds for old items with the caveat that there will be some weak effect of an earlier presentation of the old item.) This was demonstrated by taking 16 new items, determining the output of the system for each, and then determining the match between each of these outputs and the item studied last (Item 16). Then this process was repeated for the other items of the group studied last (Items 13, 14, and 15). This yielded 16 match values for each of the items in the last group. Table 8 shows the correlations across these sets of 16 match

Table 6
Match of Output From Old and New Items to the Last List Member

Learning rate	Rehearsal group size	Initial weight	No. rehearsals	Old-item match	New-item match	*SD* of new-item match
4.0	4	±0.3	2	.103	.128	.130
4.0	4	±0.3	4	.134	.199	.164
4.0	4	±0.3	8	.200	.299	.180
4.0	4	±0.3	16	.241	.266	.207
4.0	1	±0.3	2	.036	.035	.041
4.0	1	±0.3	4	.308	.337	.246
4.0	1	±0.3	8	.590	.629	.256
4.0	1	±0.3	16	.527	.584	.376
0.05	1	±0.05	2	.019	.019	.010
0.05	1	±0.05	4	.046	.045	.046
0.05	1	±0.05	8	.095	.094	.032
0.05	1	±0.05	16	.179	.176	.048
1.0	1	±0.3	2	.076	.079	.034
1.0	1	±0.3	4	.131	.137	.074
1.0	1	±0.3	8	.326	.357	.201
1.0	1	±0.3	16	.457	.539	.161

values. For the low learning rate shown in the table, there is a high correlation between Items 13, 15, and 16, and not with Item 14, indicating that Items 13, 15, and 16 formed the prototype. For example, the match for Item 16 with the output of New Item 1 was .19, with the output of New Item 2, .15, and with the output of New Item 3, .26. The corresponding matches with Item 15 show the same profile, .23, .19, and .32, respectively, and so are highly correlated. For the high learning rate, the correlations between Items 13, 15, and 16 are lower but still positive, indicating that the system was attempting to reproduce one of those last four items rather than the prototype. These correlations mirror the correlations among the four items studied last, shown in Table 9.

These results show that old–new discrimination as a function of learning can only be explained by examining several interacting factors. These include the transition from reproducing a prototype to reproducing single items in the last rehearsal group and the transition from linear to nonlinear behavior in the net input to activation transformation. The next step is to extend this investigation, moving from minor modifications of the encoder model to major changes in representational assumptions.

Effects of Similarity Among Vectors

One criticism that might be made of these simulations is that the items used in the simulations are unrelated to each other, whereas in experiments with human subjects, the items are somewhat related to each other. For example, the words in a study list are more similar to each other than they are to pic-

Table 7
Match of Output From Old and New Items to the Old and New Items

Learning rate	Rehearsal group size	Initial weight	No. rehearsals	Old-item match	New-item match	*SD* of new-item match
4.0	4	±0.3	2	.055	.004	.037
4.0	4	±0.3	4	.064	.008	.049
4.0	4	±0.3	8	.085	.017	.061
4.0	4	±0.3	16	.178	.048	.072
4.0	1	±0.3	2	.057	.007	.038
4.0	1	±0.3	4	.069	.013	.086
4.0	1	±0.3	8	.077	.027	.120
4.0	1	±0.3	16	.085	.026	.133
0.05	1	±0.05	2	.015	.000	.011
0.05	1	±0.05	4	.038	.001	.048
0.05	1	±0.05	8	.035	.001	.036
0.05	1	±0.05	16	.029	.002	.052
1.0	1	±0.3	2	.059	.011	.028
1.0	1	±0.3	4	.099	.021	.041
1.0	1	±0.3	8	.155	.040	.072
1.0	1	±0.3	16	.191	.058	.097

Table 8
Correlations (R^2) Between the Match Values for the Outputs of New Items With the Last Four List Items

	List item		
List item	16	15	14
Low learning rate ($\eta = 0.05$)			
15	+.99	—	
14	−.39	−.32	—
13	+.99	+.99	−.42
High learning rate ($\eta = 4.0$)			
15	+.74	—	
14	−.05	−.05	—
13	+.19	+.36	−.59

Note. Each item received 16 rehearsals, with four-item rehearsal groups.

tures. To examine the effect of varying the amount of similarity among items, simulations were performed in which all the items were variants on a prototype. To accomplish this, for each simulation, a random vector of zeros and ones was generated, and each of the study and test vectors in the simulation were distortions of this vector. The distortions were obtained by making some proportion of the elements of the vector different from the elements in the prototype. Various learning rates were examined, and the results were much the same as in the previous section. For low learning rates, d' as a function of number of learning trials was nonmonotonic: rising, falling, and then rising to a higher asymptote. For higher learning rates, d' rose from a low value with one learning trial to asymptote by six or eight learning trials. For Buffer Size 1, the d' function fell from an initially high value. Thus, the only way the addition of similarity among items changed the pattern of results was to give a more gradual initial rise under some high learning rate and similarity conditions.

Scaling Up Vector Length

Another potential criticism of the simulations is that a vector length of 32 is too small to be a realistic representation of stimulus information. To address this, the vector length was increased to 128 and 256 for the input and output vectors (with half that length for the hidden units). The results showed the same patterns as for the smaller vector lengths, with the absolute level of d' somewhat higher for the longer vector lengths.

More Sensitive Tests of Learning

Ratcliff, Clark, and Shiffrin (1990; see also Shiffrin, Ratcliff, & Clark, 1990) have presented a new experimental manipulation with which to evaluate models of old–new discrimination as a function of learning. Many models (Gillund & Shiffrin, 1984; Hintzman, 1986; Murdock, 1982; Pike, 1984) require that variance in the amount of match for new items increase as a function of the amount of learning of old items. When the amount of learning per item is large, then the standard deviation for new items will be higher than when the amount of learning is small. However, this prediction holds only when amount of learning is varied across lists, with all items in some lists receiving a large amount of learning and all items in other lists receiving a small amount. When amount of learning is varied for items within a list, with some items getting a large amount of learning and others only a small amount, then there can only be one value of match for new items. It follows that differences in d' (as a function of amount of learning) must be greater when amount of learning is varied within a list than when varied between lists (as long as old-item match values are comparable). This prediction that d' differences will be greater in mixed lists than in pure lists was termed the *list strength effect* by Ratcliff et al. (1990) and Shiffrin et al. (1990), and the measure used to describe the effect is a ratio of ratios: the ratio of d' values for the two conditions in the mixed list divided by the ratio of d' values for the corresponding conditions in the pure lists.

The prediction is not upheld by experimental results. With amount of presentation time per item and number of repetitions per item to manipulate amount of learning, the data from many experiments (Ratcliff et al., 1990) clearly show no differences between the mixed and pure lists; that is, the ratio of ratios was always either 1 or less than 1. Amount of learning fails to affect the difference between mixed and pure lists for recognition, even though it affects the difference for recall and even though accuracy of recognition for old test items increases with amount of study. Theoretical analyses show that some forms of some models can account for some aspects of the results, but currently these modifications are somewhat unsatisfactory (Shiffrin et al., 1990).

Because the memory models make such a strong prediction that is violated by the data, additional insight into the encoder model may be gained by examining its predictions for the mixed-list–pure-list design. To do this, an encoder model with learning rate of $\eta = 0.25$, initial weights of ± 0.3, and Buffer Size 1 was used. The study list of 16 items was divided into three groups of items: the first 6, the next 6, and the last 4. Different numbers of learning trials were given to the items in these three groups as shown in Table 10; the code represents the number of rehearsals given to each group, and the match values in the table are for the group with the slash (/) next to it. First, the main effect on performance seems to be retroactive interference. The match value for the second 6 items is almost completely determined by the number of learning trials for the last group of 4 items. For example, comparing the 4-4/-2 and 4-4/-8 conditions, one can see that the match value for the 4/ group when the last group was given two learning trials is approximately 2 times larger than the match value when the last group is given

Table 9
Correlations (R^2) Among the Last Four Input Vectors

	List item		
List item	16	15	14
15	+.36	—	
14	+.13	+.00	—
13	+.07	+.21	−.31

Note. Each item received 16 rehearsals, with four-item rehearsal groups.

Table 10
Results for the Mixed–Pure Manipulation of Amount of Rehearsal

Old-item match	New-item match	*SD* of new match	*d'*	Group type[a]
.0381	.00547	.0338	0.965	2/-2-2
.0583	.00410	.0294	1.844	2/-1-1
.0212	.00899	.0482	0.253	2/-4-4
.0251	.00636	.0410	0.457	2/-4-2
.0482	.00512	.0330	1.305	2/-1-2
.0448	.00436	.0332	1.218	2/-2-1
.0270	.00773	.0493	0.391	2/-2-4
.0584	.00547	.0338	1.566	2-2/-2
.0550	.00721	.0350	1.365	4-2/-2
.0588	.00456	.0344	1.577	1-2/-2
.0695	.00335	.0309	2.141	1-2/-1
.0682	.00436	.0332	1.923	2-2/-1
.0430	.00931	.0450	0.749	4-2/-4
.0422	.00773	.0493	0.699	2-2/-4
.0447	.0104	.0439	0.781	4/-4-4
.0766	.00721	.0350	1.983	4/-2-2
.0345	.0175	.0595	0.286	4/-8-8
.0356	.0129	.0459	0.496	4/-8-4
.0577	.00931	.0450	1.075	4/-2-4
.0526	.00796	.0382	1.168	4/-4-2
.0346	.0144	.0658	0.307	4/-4-8
.0801	.0104	.0439	1.588	4-4/-4
.0756	.0126	.0367	1.717	8-4/-4
.0790	.00899	.0482	1.452	2-4/-4
.105	.00636	.0410	2.406	2-4/-2
.100	.00796	.0382	2.401	4-4/-2
.0623	.0175	.0556	0.806	8-4/-8
.0561	.0144	.0658	0.634	4-4/-8
.0683	.0204	.0537	0.892	8/-8-8
.0859	.0126	.0367	1.997	8/-4-4
.0638	.0323	.0827	0.381	8/-16-16
.0705	.0273	.0561	0.770	8/-16-8
.0733	.0175	.0556	1.003	8/-4-8
.0703	.0155	.0411	1.333	8/-8-4
.0584	.0260	.0863	0.375	8/-8-16
.109	.0204	.0537	1.650	8-8/-8
.116	.0267	.0496	1.800	16-8/-8
.100	.0175	.0595	1.387	4-8/-8
.128	.0129	.0459	2.508	4-8/-4
.130	.0155	.0411	2.785	8-8/-4
.0915	.0332	.0806	0.723	16-8/-16
.0805	.0260	.0863	0.632	8-8/-16

[a] The group type code is as follows: The three numbers correspond to the number of learning trials for the first 6, the second 6, and the last 4 items in a 16-item list. Numbers followed by slashes indicate the group of items tested.

eight learning trials. In addition, the match values for new items are in the ratio of 2:3, and the ratio of the standard deviations in the new-item match values is also 2:3. Thus, the last set of items rehearsed is a major determinant of performance on earlier items.

These effects can be understood in terms of the way the system learns the items presented last (following from the discussion of learning effects in the prior section). When the last four items are rehearsed eight times, they dominate performance; the last item dominates most (in these examples with Buffer Size 1), and this leads to lower match values for items studied earlier. In addition, the greater the amount of learning of these items, the better the new test items will match the last two or three studied items (because, as discussed earlier, a new item will reproduce the one of the last two or three items to which it is most similar, leading to a positive match between the new item and the output). Both of these factors will serve to reduce *d'* values.

There is a small but reliable proactive interference effect that produces a positive facilitation rather than interference. This results from the use of momentum in the learning algorithm. If the initial six items receive large numbers of learning trials, then there will be a larger carryover from these learning trials to the next than if a small number of learning trials was used on the initial six items (e.g., 4-8/-8 vs. 16-8/-8). Running these simulations with no momentum eliminates the facilitation but does not qualitatively alter any of the other results.

To examine the list-strength prediction of the encoder model, the ratio of *d'* ratios (for strong and weak items) for mixed and pure lists was examined (see earlier text). It is easy to show that the encoder model predicts a large mixed-list–pure-list difference. With the results in Table 10, the conditions equivalent to those examined experimentally in Ratcliff et al. (1990) for pure lists are (for the four- and eight-repetition case) 4-4-4 and 8-8-8, and for mixed lists, 4-8-4 and 8-4-8. The ratio of strong to weak for the mixed lists is 1.76/0.66, and the ratio for pure lists is 1.27/1.19. For the pure lists, these ratios were obtained by averaging the 4/-4-4 and 4-4/-4 numbers together for the four-repetition case and 8/-8-8 and 8-8/-8 for the eight-repetition case. For the mixed lists, the weak condition is the average of 4/-8-4 and 8-4/-8 for the four-repetition case and 8/-4-8 and 4-8/-4 for the eight-repetition case. The ratio of ratios is 2.50, whereas the experimental data values are less than or equal to 1.0. Thus, the multilayer encoder model shows the same qualitative behavior as many of the other memory models and fails to predict differences in *d'* values for mixed and pure lists.

Other Parameters and Manipulations

One may think that performance of the encoder model could be improved by changes in one or another of its processing assumptions, parameter values, or representation assumptions in the if-you-only-try approach to dealing with the problems presented earlier. In this section, two of the simpler possibilities are considered; more complicated changes in architecture are examined in later sections.

One parameter of the model is the value of momentum, which is used to avoid effects of large local variations in the solution space. When a vector is entered into the system, the value of momentum is the linear combination of the newly calculated update to the system and the prior update. All of the simulations presented earlier set momentum to 0.5 (i.e., the update includes half the prior update). For simulations for which momentum was reduced to 0, there were no alterations in the qualitative behavior of *d'* as a function of amount of learning.

A second possible way to modify the system is to use some kind of sharpening process at test to enhance the output by feeding it back through the network to produce a cleaner output

(e.g., Hintzman, 1986). When this modification was added, there was no improvement in d' as the number of learning trials was varied from two to eight, although both d's were slightly higher than with unsharpened outputs. Considering the information that a test item actually retrieves from the system, one can easily see that sharpening simply gives a better reproduction of one of the last four items (assuming a high learning rate or a large number of learning trials). This result leads to a better match for the test item but little change in the pattern of d' values across conditions.

Summary for Old–New Discriminability as a Function of Learning

The behavior of d' as a function of number of learning trials is the result of several interacting factors. The value of learning rate and the initial values of weights are critical because alterations in these values can change the qualitative behavior of the d' functions. For example, low learning rates and few learning trials lead to the reproduction of a prototype of the most similar of the last few items learned. With more learning trials, the system begins to reproduce the individual vectors because weights become large, leading to nonlinearity in the net input to activation transformation. This produces a nonmonotonic function of d' as number of learning trials increases. In addition, any new test item (as well as any earlier studied old test item) will reproduce the item in the last-learned group to which it is most similar. The reproduction of this item becomes better as number of learning trials increases, and so the match value for both old and new test items increases.

The mixed–pure list manipulation that has been used to test models of memory gave the same problem for the encoder model as for the other models. The models predict larger d' differences between weak and strong items in a mixed list than they do in pure lists, contrary to data.

Autoassociative Memory Model

Before continuing to examine variants on the encoder model, I consider the autoassociative memory model proposed by McClelland and Rumelhart (1985). This model was developed specifically to account for human memory, and it is related to the encoder model because it uses the same error-correcting rule (or delta rule) for learning. It is also related to the single-layer autoassociative models proposed by J. A. Anderson (J. A. Anderson et al., 1977; Knapp & Anderson, 1984; see also Hinton, 1981) because there is only one layer of nodes, with connections from the output of each node to the input of each other node. Because the encoder model cannot correctly model human behavior in learning, it is natural to consider the autoassociative model as an alternative. It may have the same problems because it uses the same delta or error-correcting rule for learning. As with the encoder model, the autoassociative model was tested with simulations that examined the discriminability between old and new items as a function of the number of learning trials.

The autoassociative model assumes that the processing system consists of a set of nodes that are highly interconnected. Every node in the system receives both external input from outside the system and internal input from each other node except itself. Thus, the single layer of nodes serves both for input and for the final output when activation to the system has stabilized. Each node takes on activation values between 0 and 1. At input, activation values are simply the external inputs. On the first update cycle, each node passes activation to each of the other nodes multiplied by the interconnection weights, and feedback is added to the activation value from the external input to produce new activation values. After several iterations of combining external input and feedback from the network, the activation values stabilize. Then, the weights are updated by a term consisting of a constant multiplied by the activation level multiplied by the net input, where the net input is defined as the difference between the external and internal inputs. If the external and internal inputs were equal, the weight would not change. After the weight update, activation is allowed to stabilize again before another weight update, and this process continues until learning is terminated by a limit on the number of learning trials or by learning's reaching asymptote. The model was applied by McClelland and Rumelhart (1985) to data from the time course of growth of activation at test as a function of variables such as familiarity and priming condition. The model was applied in the domain of categorization and was quite successful in accounting for a range of qualitative effects.

In the simulations reported herein, discriminability between old and new items is examined as a function of the number of learning trials. The number of learning trials is assumed to correspond to the number of updates of the weights (allowing activation to stabilize between each weight update). Vector length was 16, decay and excitation multiplication constants for weight updates were both assumed to be 0.15, and the constant for weight updates was 0.05, following McClelland and Rumelhart (1985). The number of cycles used to allow activation values to stabilize before a weight update was 20. List length was assumed to be 16 items, each item a random vector with elements 1 or 0. There were 40 replications per learning condition, and means and standard errors in the means were quite stable.

Table 11 shows discrimination as a function of the number of weight updates. Essentially, discriminability decreased as the number of weight updates increased. This result is mainly due to the increase in variance in match values as the number of weight updates increased. In addition, the match for old items actually increased and then, after seven weight updates, slowly decreased, with the parameters used here and the match for new items decreased, leading to decreasing d's. In sum, the autoassociative model produces the same problematic behavior as the encoder model.

Alternative Architectures for the Multilayer Encoder Model

The architectures that were investigated in the studies so far described represent the most direct implementations of memory and learning within the multilayer and autoassociative schemes. The results show that these implementations all fail to give increases in old–new discrimination as a function of amount of training. A common response to these problems is to argue that these architectures are not the most appropriate ways of implementing memory within a distributed connec-

Table 11
Old–New Discrimination as a Function of Number of Weight Updates in the McClelland and Rumelhart (1985) Model

Weight updates	Old items		New items		
	M	*SD*	*M*	*SD*	*d'*
1	.541	.013	.498	.015	2.87
2	.565	.016	.495	.025	2.87
3	.575	.018	.498	.033	2.61
4	.579	.019	.483	.038	2.53
5	.578	.022	.477	.042	2.41
7	.574	.026	.469	.048	2.19
10	.570	.029	.464	.051	2.08
15	.568	.030	.462	.052	2.04
20	.567	.030	.462	.052	2.02

tionist framework and to suggest that there are alternative architectures that would be more consistent with prior work that has used more traditional approaches to memory. For example, some have argued that the correct way to view learning is not to assume that items are being learned for the first time but to assume that associations are being constructed between the context of a list and already-known items in permanent memory (e.g., J. R. Anderson & Bower, 1972). In the following sections, connectionist implementations of this and other schemes are examined.

Training Response Nodes

An architecture that has been used to model discrimination between two different alternatives is one in which the output node is a response node (e.g., the T–C problem, Rumelhart et al., 1986). The system is designed so that the response node will be turned on whenever one alternative is presented and turned off whenever the other alternative is presented. This scheme can be applied to recognition memory, with one difference from the usual applications: Only positive instances, and not both positive and negative instances, are trained. The reason for this difference is that in most memory experiments, it is difficult to imagine that nonstudied items could be rehearsed. To add a response node to the encoder model, there was an output layer of the same size as the input layer (31) plus one additional output node connected to each hidden unit. In training, the input vector was used as the training signal, and the response node was turned on, that is, set to 1. The aim was for the system to adjust weights so that on presentation of an input vector, the output would reproduce the input, and the single output response node would be turned on (see Figure 14).

The simple four-item rehearsal encoding scheme was used. Results were disappointing and perhaps obvious; every vector, both old and new, turned on the output node, and the average activation values for that node were near 1, and the same for old and new items. In hindsight, this result is not surprising because the system was given no information that would allow it to discriminate between positive (trained) and negative (not trained) items. Inspection of the matrix of weights provides insight into the solution obtained by the system. For the response node to have Activation 1, the net input to that node must be positive and large (so that the logistic transformation will convert it to 1). For the response node to be 0, the net input must be large and negative. Because the net input is the sum over all hidden units of the value of activation at the hidden unit multiplied by the weight (Equation 1) and because the activation values range between 0 and 1, the weight matrix values determine the sign of the net input function to the response node. If the weight values are large and positive, any hidden unit activations will produce a positive output. For the simulation of the response-node model, I found that the matrix weights from the hidden layer to the response node were all positive and large, which means that any nonzero activation at the hidden layer produced a positive value in the response node, as was observed in the simulations.

Connectionist models of discrimination (e.g., the model for the T–C problem, Rumelhart et al., 1986) appear to be successful because both positive and negative instances are trained. One might then think that the application to recognition memory could be modified by presenting negative instances at training and that this would allow old–new discrimination at test. But in psychological terms, this modification would require that when learning a list, items not in the list would also be processed with the response node set to 0. In memory procedures, at least, the assumption of rehearsal of large numbers of nonpresented items is implausible.

Item-to-Context Model

A related potential model is one in which an item is represented as a vector in the input layer and context is represented as a vector in the output layer. Then, in list learning, the set of items in the list is trained to produce a common output-context vector. At test time, old items and new items produce an output

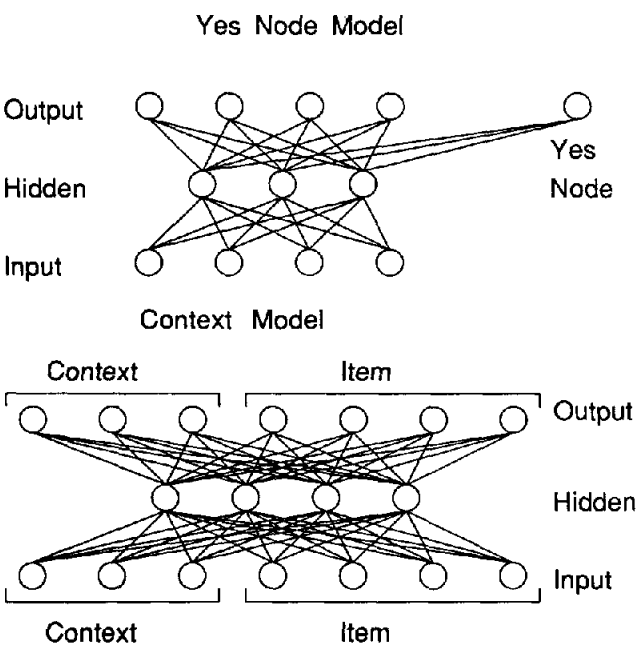

Figure 14. Illustrations of the yes node model and the context model.

that is matched to context to give a match value. For the model to work, an old-item vector should produce the context vector at test, and a new-item vector should not. The problem with this item-to-context model is exactly the same as the problem for the response-node model: The system will learn to respond with the context vector no matter what test item is presented. For a context node set to 1, the connection strengths from the hidden nodes to the context node will be set positive, and for a context node set to 0, the weights will be set negative. This will result in almost any test vector reproducing the context vector, and so result in no old–new discrimination.

New Information as Increments to Previously Learned Information

Because meaningful material is usually used in most memory procedures, it may be better to conceive of learning newly presented information as incrementing existing information in memory. In addition, the results concerning between-lists versus within-list discriminability as a function of learning (Ratcliff et al., 1990) may be addressed within this model: If memory consists of a large number of items, then the goodness of match and variability in new items will be determined mainly from contributions from items in the preexisting memory system. These contributions from preexisting items may lead to constant variability in new items as a function of amount of rehearsal of old items, leading to a prediction of no difference in the mixed–pure manipulation and thus fitting experimental results.

To implement this scheme, a standard encoder model was taught 100 random vectors, and then the system learned lists of 16 items randomly selected from the pool of prelearned vectors. The new items were other items from the set of 100 that were not selected as part of the study list on that trial. The next trial again randomly selected 16 items from the pool and could include some of the 16 items trained on the previous trial. The initial learning of the list of 100 vectors used input and output vectors of length 32 with 16 hidden units. The 100 vectors were trained for 100 sweeps of the whole pool, leading to average goodness-of-match values of approximately 0.7, which were at asymptote for learning.

In analogy to list learning, the learning process used the four-item buffer. The main variable manipulated was the number of rehearsals. Results showed a small increase in d' as a function of number of rehearsals (see Table 12), which is consistent with experimental data. However, the reason for the improvement in d' was not that goodness of match for old items improved as a function of number of rehearsals, because this quantity was constant at the value obtained in initial learning. Instead, the goodness of match of new items decreased with larger numbers of rehearsals of the old items. Thus, the model does not show better memory for old items; instead, there is more forgetting for the new items as a function of the amount of rehearsal of the old items.

This study was replicated with more resources for the system. With 100 vectors with 32 input and output elements and 16 hidden units, match was approximately 0.7, and old–new discrimination d' was approximately 0.8. With addition of many more hidden units (48 as opposed to 16), initial learning for the 100 sweeps was excellent, but there was no difference between old- and new-item match values after list learning (match for both was approximately 0.99). Thus, the system had reached almost perfect performance, so changes due to list learning produced no change in performance (cf. Table 12). Therefore, viewing list learning as retraining existing knowledge does not lead to an adequate account of memory because improvements in old–new discrimination as a function of amount of learning are the result of degradation in performance of the items that are not retrained (items that are meant to represent permanent knowledge) rather than enhancement of the items that are retrained.

Table 12
Learning as Increments to Prior Learning: Three Replications

	Old items		New items		
Sweeps	*M*	*SD*	*M*	*SD*	*d'*
		Replication 1			
1	.834	.089	.766	.108	.630
8	.841	.088	.725	.117	.991
		Replication 2			
1	.743	.183	.686	.158	.360
8	.756	.169	.635	.166	.729
		Replication 3			
1	.815	.089	.758	.117	.487
8	.813	.104	.711	.133	.767

Context and Increments to a Permanent Memory

With the next set of simulations, I attempted to model the effect of adding context to a model that is based on increments to permanent memory. I assumed that in list learning, the whole of memory is not adjusted and that only part of the vector that represents context is adjusted (see Figure 14). This is one way of implementing context-based models of memory (e.g., J. R. Anderson & Bower, 1972) within the connectionist framework.

For the context model, I assumed that part of the memory vector represented context and part represented the item, as in Figure 14. Specifically, the first 8 elements were assumed to represent context and the last 24, the item. At study, the context elements were set to (a neutral) 0.5, and the item elements were set to random numbers, either 0 or 1. One hundred random vectors were generated, and these were trained for 100 sweeps through the set of 100 vectors using learning rates of 0.25 and 4.0. This produced learning at asymptote (additional sweeps produced little change in weights). In list learning, a single initial-context vector of 8 elements was chosen (randomly selected zeros and ones). Then, 16 vectors were chosen from the set of 100 as old items, the context elements were added, and the vectors were learned with the four-item buffer described earlier. However, only the weights to and from the context elements were allowed to change during this list-learning phase, and item weights were not changed. All weights participated in produc-

Table 13
Learning for the Context Model

	Old items		New items		
Sweeps	*M*	*SD*	*M*	*SD*	*d'*
		Learning rate = 4.0			
0	.648	.058	.648	.060	.000
1	.853	.077	.837	.086	.186
2	.868	.068	.850	.081	.222
8	.885	.063	.871	.073	.192
		Learning rate = 0.25			
1	.786	.0976	.765	.0988	.213
2	.818	.0969	.796	.0989	.222
4	.841	.0946	.820	.0983	.214
8	.855	.0914	.837	.0961	.187

ing activation at the output layer. At test, the old vectors were tested along with a set of 16 new vectors (selected from that set of 100) with context elements set to the context elements used for the old items.

The main result was again that there was little change in performance as a function of number of rehearsals. Table 13 shows d' as a function of the number of sweeps and shows that d' was roughly constant at a very small value for one to eight sweeps for learning rates $\eta = 0.25$ and 4.0. In addition, the increments to discriminability from list learning were small for this context-learning scheme because item weights are responsible for much of the match so that context learning produces little differential discrimination.

This context-learning scheme seemed quite attractive because it maintains the separation of context from information about an item itself. Furthermore, learning by modifying context serves as a connectionist implementation of the traditional context model for memory. However, the results of the simulations show that the context model does not provide increases in old–new discriminability as a function of learning, and so the model is not adequate as a model of recognition memory.

D. Rumelhart and S. Sloman (personal communication, March 18, 1989) are working on a version of a context model to overcome the problem of interference in the Barnes and Underwood (1959) transfer paradigm (McCloskey & Cohen, 1989). In that paradigm, subjects were trained to criterion on an A–B list, and then forgetting on the A–B list was examined as a function of number of training trials on an A–C list. In Rumelhart and Sloman's model, the input vector has pairs of items plus context represented in a single vector (the first set of elements encodes A, the first item; the second set encodes B and C, the second item; and the third set encodes context). The output vector contains only the pair of items. For the Barnes and Underwood paradigm, all combinations of pairs of items are pretrained in a neutral context to form a permanent (perhaps semantic) memory. In the model, the pairs are trained to reproduce themselves (i.e., an autoencoder). During list learning, the A–B pairs are trained in one context in sweeps (as in the experiments), and then the A–C pairs are trained in a second orthogonal context in sweeps without repeating the A–B pairs. In addition, the learning rate on context weights is assumed to be larger (larger changes) than on item weights. With this model, interference on the A–B pairs after study of the A–C pairs is reduced.

However, this model clearly is only the beginning of an attempt to deal with the interference problem. The model will certainly have the kinds of problems with old–new discrimination that were demonstrated earlier. McCloskey and Cohen (1989) criticized an earlier version of this model on three grounds. First, they considered the combinatorial explosion involved in pretraining all possible pairs of items. Assuming a 50,000-word vocabulary, there are 2.5 billion associations to be learned. Pretraining becomes even less plausible when nonsense letter strings are considered, as used by Barnes and Underwood (1959). Although anyone can work out a relationship between any two pairs of words, this does not mean that they are preassociated in memory, and so both the representation of knowledge and the processes that are used to compute such relationships would need to be modeled, a task well beyond current modeling efforts. The second criticism is that preexperimental training of all combinations of the pairs requires concurrent training, or else the forgetting problem simply reappears during preexperimental training. It is hard to believe that all possible pairs are repeatedly available for training in a concurrent scheme. The third criticism is the assumption of orthogonal context patterns. One would like to assume that there is no abrupt change to completely different context moving from list A–B to list A–C in an experiment. Clearly, this attempt to overcome interference is just an initial step, and there are serious limitations to be addressed before any kind of comprehensive solution is found.

Another model worth contrasting with these connectionist schemes is the closed-loop model described in Murdock and Lamon (1988). This model views memory as a single-layer vector as in models by J. A. Anderson (1973) and Murdock (1982), but when an item is encoded into memory, what is encoded is the item multiplied by 1 minus the dot product of the item with the memory vector. The idea is that if the item to be stored is close to what is retrieved from memory, little is stored because information that represents that vector is already there (by storage, prototype formation, or chance). If the dot product is near 0, then nothing like the vector to be stored is encoded, and relatively large values are stored. On the face of it, this model looks like an error-correcting scheme similar to the multilayer model. However, this closed-loop model differs from the multilayer model because the multiplying factor applies equally to all nodes, whereas the backpropagation learning algorithm modifies weights to reduce error on individual nodes according to the size of the error on that node, not the average error as in the closed-loop model. Although the closed-loop model shows improvements in d' as a function of the amount of learning, it does not handle data as well as other models do (Murdock & Lamon, 1988).

Discussion

The aim of the research described in this article was to develop, examine, and test a set of connectionist models for recognition memory. I initially hoped to develop a competitive model

to the currently successful models of memory (e.g., Gillund & Shiffrin, 1984; Hintzman, 1986; Murdock, 1982). The results are, on the one hand, discouraging because they have not produced a model that can handle the most central phenomena of recognition memory: learning and forgetting. On the other hand, the results are encouraging because they show general limitations on the class of connectionist models that was examined. The more obvious models can be falsified, so there exist constraints on the behavior of the models that seem central to their structure.

The models I have considered differ from prototypical connectionist models (of the kind used by McClelland & Rumelhart, 1985) because items are learned one at a time or in small groups, not all at once. I argue that this sequential presentation is the way traditional list-learning procedures must be modeled because remote rehearsal probably cannot be used, especially at fast presentation rates, and because the models provide no mechanism for recovery of earlier items for later rehearsal. With sequential presentation, the main problems with the models are, first, that there is rapid forgetting of extremely well-learned information because of learning of subsequent information and second, that discrimination between (studied) old items and (not studied) new items decreases to asymptote as a function of the amount of training on the old items.

Forgetting was examined first with small four-element orthogonal vectors that served as study and test items. A subset of these vectors was trained to a high criterion (corresponding to almost perfect reproduction), and then forgetting was examined as a function of training another subset to a high criterion. Results showed that the output produced when one of the earlier set of vectors was tested was a blend of both the old vector and the vector learned last. In fact, sometimes the output in response to one of the earlier items actually matched the last-studied item best (i.e., the model only retrieved the last item presented). Several manipulations were carried out to try to correct this problem. First, only smaller weights were adjusted, and second, extra resources were provided to the network by adding hidden units. Both of these schemes helped the system somewhat to reproduce more accurately earlier learned vectors, but there was still considerable interference from subsequently learned items, and no scheme appeared capable of adequately overcoming this problem.

For larger vectors with randomly selected elements (1 or 0), the same kinds of results were obtained. There was rapid reduction in the match between an input vector and the output of the system as a function of intervening learning. For items learned in groups, there was less forgetting than for items learned singly. However, forgetting even for a group was large, and this was true even after only one additional item was trained to criterion.

These results speak to a more general issue that has been raised for this class of error-correcting models, and that is their instability in a variable training environment (Grossberg, 1987; McCloskey & Cohen, 1989). Grossberg has forcefully argued that many of the recent network models are unstable under temporally changing training environments, and the problem is illustrated with reference to the backpropagation learning algorithm. The results presented here demonstrate that these effects apply even to smaller domains such as learning a list of items. In addition, the results I have presented provide a quantitative account of the size of the effects. The results show that the system tracks the item presented last (any test item will reproduce it) with some memory for earlier information superimposed on this output (see Figure 7). This finding suggests that the error-correcting backpropagation scheme may not be a good model of human memory for recently presented information. However, in conditions in which the training environment is stable (on average), such as in early visual experience, in early reading, or in early auditory experience, the environment may be stable enough for the model to apply (but my results should be taken as a warning that the system should be tuned so that it does not track just the latest information presented at the expense of other information).

The next set of studies examined discrimination between old vectors (those trained in the sequential list-learning procedure) and new vectors that were not trained. The result that was most disturbing was that old–new discrimination decreased or was nonmonotonic as the amount of learning for each item increased until an asymptote in old–new discrimination was reached. This result clearly is contradicted by data from recognition memory studies that all show increasing d' as a function of the amount of learning. A summary of the models and their problems is shown in Table 14.

An interesting contrast worth reviewing concerns the connectionist models that use error-correcting schemes and the linear models of J. A. Anderson (1973) and Murdock (1982) for recognition. In these latter models, items are vectors, and an item is stored in a common memory vector by adding each element to the corresponding element in the memory vector. These linear models have some of the same problems as the multilayer model described earlier. When a set of vectors is learned, the models produce a constant d' as a function of the number of repetitions of the vectors. This result occurs because although the mean match of old items increases, the standard deviation in the noise increases in the same ratio to keep discriminability constant. The behavior of the multilayer connectionist model is somewhat similar. For small numbers of rehearsals, the mean match for studied items increases, but the match for new items and the standard deviation in new-item match increase more rapidly, leading to a drop in d' or constant d'. Although the behavior of these models is not quite the same, the same problem is indicated: the increase in noise as a function of the amount of learning for old items. Although increases in noise are built in to several of the global memory models, there are schemes that reduce the effect so that discrimination increases as a function of learning (see Gillund & Shiffrin, 1984; Hintzman, 1986; Murdock, 1989). However, as noted earlier, these approaches all have problems because data from mixed–pure list design experiments show that noise either does not increase as a function of the amount of learning or that the mean match for old and new items in a mixed list design is different than in a pure list design, thereby canceling the increase in noise (Ratcliff et al., 1990; Shiffrin et al., 1990).

Several alternative models were considered that embodied theoretical constructs often used in theories of memory. For example, first, a model was examined that assumed that in list learning a response node was trained to signal that a test item had been studied. This model failed because old–new discriminative information was not provided at training, and so the re-

Table 14
Summary of Some of the Models Tried

Model	Description
Backpropagation	Items are random vectors; sequential learning; an item is trained to reproduce itself at output. *Problems:* rapid forgetting; old–new discrimination decreases to a constant level or is nonmonotonic as a function of learning; mixed versus pure list *d*'s.
Yes node using backpropagation	In training, a single node (yes node) is turned on for each item trained. For each pattern learned, the yes node is turned on. *Problem:* At test it is turned on for all patterns; this same problem would occur for any multilayer model that used a common context with item-to-context links being trained.
Semantic memory	Learns 100 vectors to criterion prior to list learning; list learning then selects 16, retrains them, and tests that subset (old items) versus another randomly selected subset (new items). *Problem:* At test, old items retain their match value as a function of presentation time, new match values are lower the more the old subset is retrained, i.e., increases in d' are due to forgetting of new items.
Context model	One fourth of the vector designated context; trains a semantic memory of the noncontext elements; at list learning, modifies only the context–hidden–context elements. *Problem: d'* increased from no list learning to some but was constant from then on.
Autoassociative model	*Problem:* Same problems with lack of improvement in old–new discriminability as a function of learning.

sponse node was active for all test vectors, both old and new. Second, a model in which list learning was viewed as incrementing preexisting memory had the problem that list learning did not increase the match for old items but rather produced forgetting in the items not rehearsed in the study list. The problem with this result is that learning should result from strengthening of old items, not forgetting of untrained permanent knowledge. Third, a model that used a part of the input and output vectors as context failed because old–new discrimination was constant as a function of learning after one learning trial.

These problems, derived from a range of models, provide constraints on potential connectionist models of memory. Furthermore, these results can serve as benchmarks for future connectionist models of memory by providing phenomena that need to be addressed as a first step in model building. There are classes of models not addressed in these studies, models using the Hebb rule for learning and others that do not use the error-correcting scheme (e.g., J. A. Anderson, et al., 1977; Carpenter & Grossberg, 1987; Kosko, 1987). An interesting conjecture based on these results is that the problems arise from the use of the error-correcting scheme and distributed representations. Further work will be needed to demonstrate that this is true and that other classes of learning rules do not have these problems.

The issue of how information is represented has not been addressed in these simulations and models. The issue of representation is important and could influence the kinds of models that will be developed in the future. These models will have to take into account representational constraints as well as the kinds of results I have presented. The rationale for use of random vectors in these simulations and not structured vectors is common to many memory models. The models assume a homogeneous set of items in which individual items are no more or less related to each other on average. Many of the successes of the connectionist program use assumptions about the featural representation of the stimulus information. However, little systematic progress has been made in developing general principles for the selection of representations for use in these connectionist models. When a choice of representation has been made, the system can abstract from the desired input and output relations to obtain an internal mapping between these two (sometimes with interpretable hidden units). However, there is no discovery procedure for the representation of input and output features. For lower level perceptual and motor processes, guidance from the kinds of representations and transformations that these systems use may be obtained. However, such an approach is necessarily limited by knowledge of the physiological system, and this is currently available only for the more peripheral processing systems.

The main conclusion to draw from this work is that the obvious connectionist schemes for memory that use multilayer models have severe problems. The results presented in this article provide a challenge for connectionist models and a set of criteria that must be reached if connectionist models are to compare with the established global models of memory.

References

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

Anderson, J. A. (1972). A simple neural network generating an interactive memory. *Mathematical Biosciences, 14,* 197–220.

Anderson, J. A. (1973). A theory for the recognition of items from short memorized lists. *Psychological Review, 80,* 417–438.

Anderson, J. A., & Hinton, G. E. (1981). Models of information processing in the brain. In G. E. Hinton & J. A. Anderson (Eds.), *Parallel models of associative memory.* Hillsdale, NJ: Erlbaum.

Anderson, J. A., Silverstein, J. W., Ritz, S. A., & Jones, R. S. (1977). Distinctive features, categorical perception, and probability learning: Some applications of a neural model. *Psychological Review, 84,* 413–451.

Anderson, J. R., & Bower, G. H. (1972). Recognition and retrieval processes in free recall. *Psychological Review, 79,* 97–123.

Atkinson, R., & Shiffrin, R. (1968). Human memory: A proposed system and its control processes. In K. Spence & J. Spence (Eds.), *The psychology of learning and motivation* (Vol. 2, pp. 90–195). New York: Academic Press.

Barnes, J. M., & Underwood, B. J. (1959). Fate of first-list associations in transfer theory. *Journal of Experimental Psychology, 58,* 97–105.

Carpenter, G. A., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision, Graphics, and Image Processing, 37,* 54–115.

Eich, J. M. (1982). A composite holographic associative recall model. *Psychological Review, 89,* 627–661.

Eich, J. M. (1985). Levels of processing, encoding specificity, elaboration, and CHARM. *Psychological Review, 92,* 1–38.

Gillund, G., & Shiffrin, R. M. (1984). A retrieval model for both recognition and recall. *Psychological Review, 91,* 1–67.

Grossberg, S. (1980). How does a brain build a cognitive code? *Psychological Review, 87,* 1–51.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science, 11,* 23–63.

Grossberg, S. (1988). Nonlinear neural networks: Principles, mechanisms, and architectures. *Neural Networks, 1,* 17–61.

Hinton, G. E. (1981). Implementing semantic networks in parallel hardware. In G. E. Hinton & J. A. Anderson (Eds.), *Parallel models of associative memory.* Hillsdale, NJ: Erlbaum.

Hintzman, D. L. (1986). "Schema abstraction" in a multiple-trace memory model. *Psychological Review, 93,* 411–428.

Hintzman, D. L. (1988). Judgments of frequency and recognition memory in a multiple-trace memory model. *Psychological Review, 95,* 528–551.

Knapp, A. G., & Anderson, J. A. (1984). Theory of categorization based on distributed memory storage. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 10,* 616–637.

Kohonen, T. (1978). *Associative memory: A system-theoretical approach.* Berlin, Federal Republic of Germany: Springer-Verlag.

Kosko, B. (1987). Adaptive bidirectional associative memories. *Applied Optics, 26,* 4947–4960.

McClelland, J. L., & Rumelhart, D. E. (1985). Distributed memory and the representation of general and specific information. *Journal of Experimental Psychology: General, 114,* 159–188.

McCloskey, M., & Cohen, N. J. (1989). Catastrophic interference in connectionist networks: The sequential learning problem. In G. H. Bower (Ed.), *The psychology of learning and motivation.* New York: Academic Press.

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

Murdock, B. B., Jr. (1982). A theory for the storage and retrieval of item and associative information. *Psychological Review, 89,* 609–626.

Murdock, B. B., Jr. (1983). A distributed memory model for serial-order information. *Psychological Review, 90,* 316–338.

Murdock, B. B., Jr. (1989). Learning in a distributed memory model. In C. Izawa (Ed.), *Current issues in cognitive processes: The Tulane Floweree Symposium on Cognition.* Hillsdale, NJ: Erlbaum.

Murdock, B. B., Jr., & Lamon, M. (1988). The replacement effect: Repeating some items while replacing others. *Memory & Cognition, 16,* 91–101.

Pavel, M., & Moore, R. T. (1988). *Computational analysis of solutions of two-layer adaptive networks.* Unpublished manuscript.

Pike, R. (1984). Comparison of convolution and matrix distributed memory systems for associative recall and recognition. *Psychological Review, 91,* 281–294.

Ratcliff, R. (1978). A theory of memory retrieval. *Psychological Review, 85,* 59–108.

Ratcliff, R. (1988). Continuous versus discrete information processing: Modeling the accumulation of partial information. *Psychological Review, 95,* 238–255.

Ratcliff, R., Clark, S. E., & Shiffrin, R. M. (1990). The list-strength effect: I. Data and discussion. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16,* 163–178.

Ratcliff, R., & McKoon, G. (1988). A retrieval theory of priming in memory. *Psychological Review, 95,* 385–408.

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning integral representations by error propagation. In D. E. Rumelhart & J. L. McClelland, *Parallel distributed processing: Explorations in the microstructures of cognition. Vol. 1: Foundations.* Cambridge, MA: MIT Press.

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing: Explorations in the microstructures of cognition. Vol. 1: Foundations.* Cambridge, MA: MIT Press.

Rundus, D., & Atkinson, R. C. (1970). Rehearsal process in free recall: A procedure for direct observation. *Journal of Verbal Learning and Verbal Behavior, 9,* 99–105.

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

Shiffrin, R. M., Ratcliff, R., & Clark, S. E. (1990). The list-strength effect: II. Theoretical mechanisms. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16,* 179–195.

Received February 27, 1989
Revision received September 5, 1989
Accepted September 26, 1989 ■