COGNITIVE SCIENCE **16**, 41–79 (1992)

Connectionist and Memory-Array Models of Artificial Grammar Learning

ZOLTAN DIENES
University of Oxford

Subjects exposed to strings of letters generated by a finite state grammar can later classify grammatical and nongrammatical test strings, even though they cannot adequately say what the rules of the grammar are (e.g., Reber, 1989). The MINERVA 2 (Hintzman, 1986) and Medin and Schaffer (1978) memory-array models and a number of connectionist autoassociator models are tested against experimental data by deriving mainly parameter-free predictions from the models of the rank order of classification difficulty of test strings. The importance of different assumptions regarding the coding of features (How should the absence of a feature be coded? Should single letters or digrams be coded?), the learning rule used (Hebb rule vs. delta rule), and the connectivity (Should features be predicted only by previous features in the string, or by all features simultaneously?) is investigated by determining the performance of the models with and without each assumption. Only one class of connectionist model (the simultaneous delta rule) passes all the tests. It is shown that this class of model can be regarded by abstracting a set of representative but incomplete rules of the grammar.

Recently, there has been considerable interest in how subjects learn artificial grammars (Dienes, Broadbent, & Berry, 1991; Mathews et al., 1989; Perruchet & Pacteau, 1990; Reber, 1967, 1976, 1989; Servan-Schreiber & Anderson, 1990). The complex but specifiable stimulus structures generated by the grammars provide ideal test cases for theories of human learning (Reber, 1989). In a typical experiment subjects memorize strings of letters that appear arbitrary but are actually generated by a finite state grammar. Figure 1 shows a typical finite state grammar. Subjects are then informed of the existence of the complex set of rules that constrains letter order (but not what the rules are), and are asked to classify new grammatical and nongrammatical strings. Subjects' typical classification performance—about 70%—indicates that they have acquired substantial knowledge about the

This research was supported by the Economic and Social Research Council. I gratefully acknowledge Gerry Altmann, Dianne Berry, Donald Broadbent, Vernon Dobson, James McClelland, Arthur Reber, Alexandros Trevis, and an anonymous reviewer for valuable comments. I am also grateful to Don Dulany for making his raw data available.

Correspondence and requests for reprints should be sent to Zoltan Dienes, now at the School of Experimental Psychology, University of Sussex, Brighton, Sussex, BN1 9QG, England.

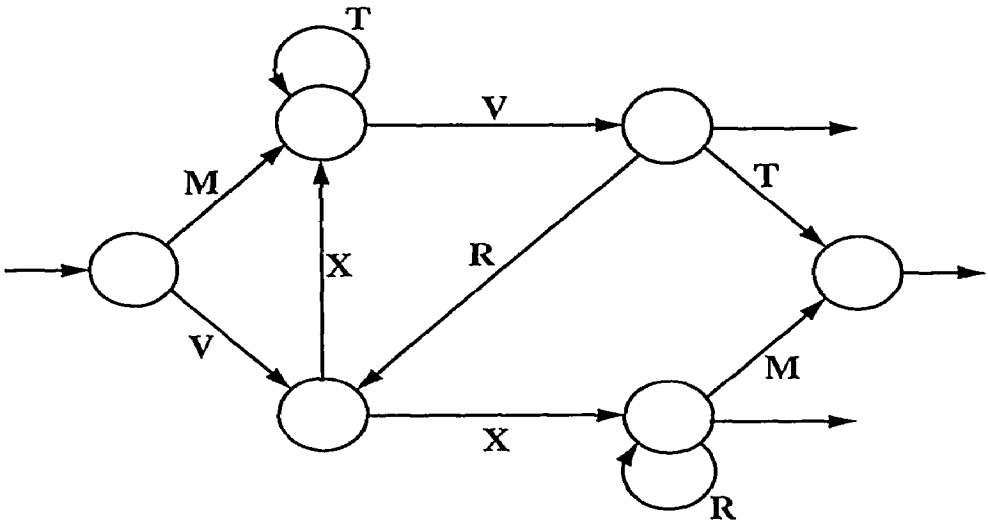

Figure 1. Example of a finite state grammar

grammar. This knowledge is *implicit* in that the learning occurs incidentally (Mathews et al., 1989; Reber, 1976) and subjects are unable to justify adequately their classification decisions in free report (Dienes et al., 1991; Mathews et al., 1989; Reber & Allen, 1978). These findings suggest that subjects do not learn the grammars by deliberate hypothesis testing.

Two currently influential approaches to modelling learning do not involve hypothesis testing and have yet to be systematically applied to grammar learning: The connectionist and memory-array approaches. In the former case, lawful behavior may be produced by a connectionist network in which rules or hypotheses are not explicitly represented, but emerge from the way that interacting units are connected (Rumelhart & McClelland, 1986a). In the case of the memory-array approach (e.g., Estes, 1986; Hintzman, 1986; Medin & Schaffer, 1978), rules are also not explicitly represented but emerge from the way in which test items are compared to stored exemplars.

Both approaches have been applied to human concept formation, as will be discussed later, but there is a need for more systematic testing of both types of models against empirical data. In terms of the connectionist approach, Massaro (1988) argued that specific models have rarely been tested against plausible alternatives so as to indicate which assumptions in a model are necessary and which are extraneous. And in terms of the memory-array approach, the mathematical models that have evolved have not been applied to the artificial grammar-learning paradigm, even though the approach originally gained impetus by its implications for artificial grammar learning (Brooks, 1978).

TABLE 1
Strings Presented in Acquisition and Test Periods

Acquisition	Test	
Grammatical	Grammatical	Nongrammatical
MTTTTV	VXTTTV	VXRRT
MTTVT	MTTTV	VXX
MTV	MTTVRX	VXRVM
MTVRX	MVRXVT	XVRXRR
MTVRXM	MTVRXV	XTTTTV
MVRX	MTVRXR	MTVV
MVRXRR	MVRXM	MMVRX
MVRXTV	VXVRXR	MVRTR
MVRXV	MTTTVT	MTRVRX
MVRXVT	VXRM	TTVT
VXM	MVT	MTTVTR
VXRR	MTVT	TVTTXV
VXRRM	MTTV	RVT
VXRRRR	MVRXR	MXVT
VXTTVT	VXRRR	VRRRM
VXTVRX	VXTV	XRVXV
VXTVT	VXR	VVXRM
VXVRX	VXVT	VXRT
VXVRXV	MTV	MTRV
VXVT	VXRRRM	VXMRXV
	VXTTV	MTM
	VXV	TXRRM
	VXVRX	MXVRXM
	VXVRXV	MTVRTR
	MVRXRM	RRRXV

This article investigates the usefulness of both the connectionist and memory-array approaches in modelling artificial grammar learning. Initially, the experimental data used to evaluate the models is described, and then the models used are described. Next, the models are evaluated against the empirical data. Finally, characteristics of the most successful model are explored in more detail. This section attempts a higher level description of the successful model in order that a deeper understanding may be obtained of the sort of knowledge acquired by it.

THE EXPERIMENTAL DATA

The data used to evaluate the models were obtained from Dienes et al. (1991) and Dulany, Carlson, and Dewey (1984). In both studies, subjects were initially exposed to the 20 acquisition strings shown in Table 1, and were asked to memorize them. Then subjects were asked to classify the 25

TABLE 2
Mean Experimental Data

	Pc	CC	CE	EE
Dienes et al. (1991)	.63	.48	.15	.22
Dulany et al. (1984)	.63	.51	.15	.20

Note. Pc is the proportion of strings correctly classified; CC is the proportion correctly classified twice in a row; CE is the proportion correctly classified once correctly and once in error; and EE is the proportion classified in error twice in a row.

grammatical and 25 nongrammatical test strings shown in Table 1. Subjects saw each string twice, and therefore made 100 classification decisions. Dienes et al. provided data from eight separate subject groups, with a total of 82 subjects, and Dulany et al. provided data from four separate subject groups, with a total of 50 subjects.[1]

The experiments provided data on the average classification performance of subjects (Pc), the range of classification performance, the proportion of strings classified correctly twice (CC), the proportion classified once correctly and once in error (CE), and the proportion classified in error twice (EE). The experiments also provided data on the rank order of string difficulty for both grammatical and nongrammatical strings.

The average Pc, CC, CE, and EE values for Dienes et al. (1991) and Dulany et al. (1984) are shown in Table 2. Although the average values happen to be very similar across the two studies, the precise values vary depending on experimental condition. The range of individual subject scores is also quite large; in Dienes et al., Pc varied between chance and 80%. Thus, an adequate model should be able to produce Pc values near 80%.

Another constraint on the models concerns the relationship between CE and EE. Reber (e.g., 1989) emphasized the theoretical importance of this relationship; specifically, Reber argued that if the subjects' CE and EE are

[1] The groups in Dienes et al. (1991) and Dulany et al. (1984) differed at the learning stage. Some groups of subjects were asked to search for rules, other groups were not informed of the existence of rules; both Dienes et al. and Dulany et al. found that this had no influence on performance. Some groups tested by Dienes et al. were asked to perform a concurrent secondary task (random number generation), other groups were not; Dienes et al. found that concurrent random number generation deteriorated performance. Some groups tested by Dulany et al. were presented the learning strings sequentially, others simultaneously; this did not affect performance. Because there is no evidence that the experimental manipulations affected performance qualitatively on the test strings, their details will not be important for the modelling conducted in this article. Dienes et al. also included a group (the mixed group) exposed to nongrammatical as well as grammatical strings of learning. Because this group was exposed to a different set of strings than the other subjects, and the models, they are not considered in this article.

very similar, then their knowledge can be regarded as representative of the grammar. Consider a subject who correctly knows a certain proportion of the strings and guesses for the remaining strings. Then any string that was in error once or twice (CE or EE) must have been classified simply by guessing. Thus, with the probability of a correct guess set at .50, CE would equal EE. However, if the subject systematically misclassifies some strings because of rules not representative of the grammar, then EE would be greater than CE. Dienes et al. (1991) and Dulany et al. (1984) consistently found a slightly but significantly greater EE than CE value. Reber also often obtained values similar to the means displayed in Table 2 (see Dulany et al., 1984, p. 546 for discussion on this point). The important aspect of these results, which will be used to constrain the models, is that EE should be slightly, but only slightly, greater than CE; the difference EE − CE was taken to be .05. Thus, a model that could obtain high enough Pc values only by inflating EE considerably above CE, would not be adequate.

The experiments provided data on rank orderings of string difficulty and these were used to assess the adequacy of different models. It is important that the rank orderings represent a reliable aspect of subject performance. The obtained rank orderings may, for example, simply represent random scatter. Indeed, Mathews et al. (1989) emphasized the divergence between knowledge representations of different subjects, as indexed by the contents of free-recall reports. However, the extent of the overlap across subjects in the knowledge representations underlying classification performance remains an open question.

To assess the reliability of the experimental rank orderings, separate rank orderings were determined for the eight groups in Dienes et al. (1991). Rank orderings were determined by summing the number of correct responses to each string. The 28 Spearman's correlations between each group and all the other groups were transformed by Fisher's z and averaged. The mean Fisher's z converted back to a correlation was .68 for grammatical exemplars (with a standard deviation, SD_{n-1}, in z scores of .19), and .52 for nongrammatical exemplars ($SD_{n-1} = .17$). Thus, there is considerable overlap between the groups in which strings were found difficult. The agreement among subjects in all the Dienes et al. groups can also be indexed by Cronbach's alpha: for grammatical exemplars, $\alpha = .56$, and for nongrammatical exemplars, $\alpha = .65$.

The data from the subjects run in Dienes et al. (in press) were combined to provide a single rank ordering (call it RANK1). The Spearman's correlation between RANK1 and the rank ordering derived from Dulany et al.'s (1984) data[2] was .68 for grammatical exemplars, and .59 for nongrammatical exemplars, $ps < .01$.

[2] Many thanks to Don Dulany for making these data available.

In summary, there was considerable consistency in the rank ordering of exemplar difficulty between subjects. An adequate model of artificial grammar learning should be able to account for this consistency.

The data from Dienes et al. (in press) and Dulany et al. (1984) were combined to produce a single rank ordering (call it TOTAL) to test the models with. It may be objected that, by averaging over all subjects, a rank ordering is obtained that is not representative of any single subject. Although recognizing this possibility, the simple assumption will be made here, based on the high levels of between-subjects consistency found earlier, that the obtained average rank ordering represents a central tendency, around which each subject deviated by a random error.

DESCRIPTION OF MODELS

The two types of model considered, connectionist models and the memory-array models of Estes (1986; also, Medin & Schaffer, 1978) and of Hintzman (1986), are now described in turn.

Connectionist Models

Initially, the relevant ideas in connectionism are introduced, and then two influential learning rules are explored in more detail. Next, previous experimental applications of connectionist ideas to human concept formation are briefly reviewed, and then the details of the connectionist models used in this article are described.

Introduction. Connectionism is having an increasing impact on psychology; indeed, Massaro (1988) called it a revolution, and Schneider (1987), a "paradigm shift." Connectionism attempts to model human performance according to patterns of activation across a number of simple computational elements, or units, connected by weights. The architecture of a network of units is specified by the connectivity between the units, that is, which weights are allowed to be nonzero. For example, the *autoassociator* can be represented by a set of units all connected to each other (but no connections to the same unit from itself); see Figure 2. McClelland and Rumelhart (1986) argued that the autoassociator could provide a useful model of human learning and concept formation.

A stimulus presented to an autoassociator produces a pattern of activation (the *input* activation) across the units at time t. At time $t+1$, the autoassociator produces a response: An *output* activation for each unit based on the weighted sum of the input activations of the other units. The aim of the autoassociator is to produce output activation equal to the input activation. The simplest output function, and the one used for the models in this article, is

$$o_i = \sum_j w_{ij} a_j$$

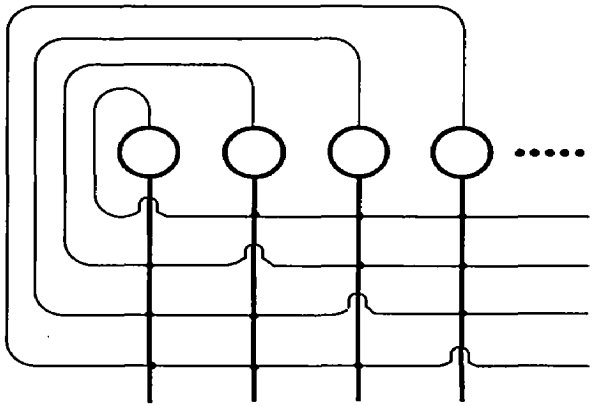

Figure 2. An autoassociator

where o_i is the output activation of the ith unit, w_{ij} is the weight from the jth unit to the ith unit, and a_j is the input activation of the jth unit. Models with hidden units require more complex output functions. In order to investigate the performance of the simplest models, this article does not use models with hidden units.

In a feed-forward autoassociator, activation passes through the weights just once to produce the output activation. In a recurrent autoassociator, the output activation arriving back at each node can be passed through the weights again until a stable state is reached. This article will mainly investigate feed-forward rather than recurrent autoassociators. When a recurrent autoassociator is used, it will be explicitly labelled as such.

The autoassociator is interesting from the point of view of artificial grammar learning because the task of forming suitable weights in an autoassociator is similar to the task of subjects learning the grammar: that is, to establish the predictability of each letter from the other letters in a grammatical string. For example, a given input vector **a** could represent a particular string. The different units could represent letters in different positions.

Part of the appeal of connectionist networks is that there is often no need to set the weights by hand in order to produce appropriate behavior; the network can learn to "program itself" by the use of local learning rules. Two rules that are commonly used in networks without hidden units and that have been suggested as models of human learning are the Hebb rule (Anderson, 1983) and the delta rule (Rumelhart & McClelland, 1986b). These rules are discussed in turn.

(a) The Hebb Rule. The Hebb rule is so called because it was clearly espoused by Hebb (1949), although a previous statement of it appeared in James (1890/1950) as his "law of neural habit." The Hebb rule is that the

increment in weight between two units is dependent on the correlation between the activations of the two units. A common version of this rule (see, e.g., Anderson, 1983), and the simplest version, is for the weight to be incremented on each learning trial by an amount equal to the product of the activations of the two units. That is,

$$\Delta w_{ij} = a_i a_j$$

where a_i and a_j are the activations of the ith and jth units, respectively.[3] Thus, the weight between units i and j will be increased only if both units are on (have positive activations) during the learning trial.

(b) The Delta Rule. Whereas the Hebb rule was developed as a plausible way in which neurons might learn (Hebb, 1949; James, 1890/1950), the delta rule was developed as an optimal solution to a computational problem (see Hinton, 1987; Stone, 1986; Widrow & Hoff, 1960). The delta rule is the procedure that, by gradient descent, will produce a set of weights that minimizes the squared difference between the desired output vectors and the actual output vectors produced by the network. It can be shown (e.g., Hinton, 1987) that this implies the delta rule

$$\Delta w_{i,j} = \Sigma_k \, LR(d_{i,k} - o_{i,k})a_{j,k}$$

where $d_{i,k}$ is the desired output for pattern k of the ith unit, and $o_{i,k}$ is the actual output, if the output of each unit is simply the weighted sum of its input. LR is a sufficiently small learning rate. In the case of an autoassociator, the desired output is the input, that is, $d_{i,k} = a_{i,k}$.

Consider an autoassociator trained according to either the Hebb or the delta rule. Initially, in a *learning phase,* a set of input activation vectors (call the kth such vector $\mathbf{a}_k$) are successively applied. Each learning vector could represent, for example, a particular string generated by a finite state grammar. After each vector, the weights matrix, $\mathbf{W}$, is changed according to the learning rule. Then, in a *testing phase,* the ability of the autoassociator to predict each activation value of a test vector $\mathbf{a}_k$ can be determined. The vector of predicted (or output) activations for the kth test vector is given by

$$\mathbf{p}_k = \mathbf{W}\mathbf{a}_k.$$

[3] With this version of the Hebb rule, w_{ij} can increase without limit. This is not a problem for the use of the Hebb rule in this article, but a learning rule might appear more natural if it leads to stable w_{ij} after some period of exposure to an ergodic sequence. The following rule could be used for this purpose: $w_{i,j,n+1} = w_{i,j,n} + LR(a_i a_j - w_{i,j,n})$, where $w_{i,j,n}$ is the weight between the ith and jth units on trial n, and LR is a small learning rate; $w_{i,j}$ will be stable over a long sequence of trials when it equals the expected value of $a_i a_j$ over trials. Note that the pattern of $w_{i,j}$ produced by this rule after a sufficiently large number of trials will equal the pattern produced by the simpler rule given in the text after a single iteration through all strings. Thus, the simpler rule is actually used in this article to model grammar learning.

As will be explained in the next section, if $\mathbf{a}_k$ is close to a "central tendency" extracted by the autoassociator, then $\mathbf{p}_k$ will closely match $\mathbf{a}_k$. The degree of match can be assessed by the correlation between $\mathbf{a}_k$ and $\mathbf{p}_k$ (specifically, by the cosine of the angle between $\mathbf{a}_k$ and $\mathbf{p}_k$). For example, the autoassociator may have been trained on grammatical strings in the learning phase. If it had learned the constraints that apply between letters in a grammatical string, it would predict each activation value for vector $\mathbf{a}_k$ more accurately if $\mathbf{a}_k$ represented a grammatical rather than nongrammatical string. Thus, the variation in the correlation between $\mathbf{a}_k$ and $\mathbf{p}_k$ could be used by the autoassociator to classify grammatical and nongrammatical strings.

If the **W** produced by a Hebb or delta rule produced a similar pattern of classification performance as produced by subjects, how would our understanding of artificial grammar learning (and hence, learning in general) be increased? Understanding the type of concept acquired by the autoassociators is essential if our understanding of what the associator is learning is to go beyond a simple enumeration of its weights. The next section attempts to characterize the sort of knowledge acquired by Hebb and Delta rule autoassociators.

Nature of the Hebb and Delta Rules

(a) The Hebb Rule. The *i, j*th entry of the **W** of a Hebbian autoassociator reflects how frequently units *i* and *j* both had positive activations or both had negative activations. That is, **W** constitutes an approximate sample covariance matrix for the units (Anderson, 1983). The eigenvectors[4] of a correlation matrix give the principal components of principal components analysis (PCA). Thus, the principal eigenvectors of **W** will contain appreciable loadings for features that are mutually highly correlated. Extracting this underlying structure in the patterns may be regarded as a form of concept formation. Indeed, Child (1970), who wrote a book on factor analysis, compared extracting factors to a child forming concepts.

A key difference between **W** and the correlation matrix used by PCA is that the entries in **W** are strictly covariances only if the mean activation of each unit over learning trials is zero. The entries in **W** are correlations only with the further requirement that the standard deviations of the activations of each unit are 1. These requirements are not easily met in a natural coding scheme. For example, a readily interpretable coding scheme is for a unit to have an activation of 1 if the feature it codes (e.g., a particular letter in a

[4] $\mathbf{a}$ is an eigenvector of matrix $\mathbf{M}$ if $\mathbf{Ma} = \lambda\mathbf{a}$, where λ is a constant (called the eigenvalue). Thus, determining the eigenvectors of the weights matrix **W** of an autoassociator amounts to determining which vectors the autoassociator will successfully complete (to a scalar multiple). Characterizing the eigenvectors of **W** amounts to characterizing the sort of knowledge acquired by the autoassociator.

particular position) is present in an exemplar, and an activation of 0 (or -1) otherwise. The entries in **W** will then reflect not only the covariance between the activations of two units, but also their frequency of occurrence. Thus, a unit will be strongly represented in the dominant eigenvector not only if its activation correlates highly with the activations of other units (as in PCA), but also if it has a high base rate of occurrence. This difference to PCA is desirable if the autoassociator is to be sensitive to base rate effects.

By analogy with PCA, if there are only a few eigenvectors of **W** with appreciable eigenvalues, these principal eigenvectors may be regarded as having extracted the "central tendencies" of the exposed exemplars. In this sense, the Hebbian autoassociator may be regarded as having learned a "concept." But how might this knowledge of the concept actually be expressed?

The vector of output activations across all units for the kth input pattern is given by

$$\mathbf{o}_k = \mathbf{W}\mathbf{a}_k.$$

If the exemplars learned by a Hebbian auto associator are coded as orthogonal vectors, then the auto associator will be able to reproduce each input vector entirely (to a scalar multiple) without interference from the others. That is, the $\mathbf{a}_k$ will form the eigenvectors of **W**. In the artificial grammar-learning task, the exemplars possess a strong family resemblance structure defined by the finite state grammar. Any scheme for coding the exemplars that captures this family resemblance structure must represent the exemplars in a nonorthogonal way. Thus, with such a coding scheme, there is likely to be a dominant eigenvector that almost entirely captures the variance in all the exposed exemplars. Test strings that are highly "prototypical," that is, close to the dominant eigenvector, will be little changed (to a scalar multiple) by multiplication by **W**. That is, the correlation between input and output activations will be close to 1. If, however, a test string is not close to the dominant eigenvector, the output vector will nonetheless be pulled towards the eigenvector,[5] and so the correlation between input and output vectors will be somewhat less than 1. The variation in the correlation between input and output vectors can be used as a means for classifying test strings as "grammatical" or not.

To summarize, a Hebbian auto associator provides a measure of the central tendencies of a set of exemplars in terms of PCA. Test strings can be classified according to how well they match the "principal components" of the studied exemplars, where the "principal components" have been modified by base rate effects.

(b) The Delta Rule. The delta rule is an iterative method of producing the standard regression coefficients for predicting each unit from the other

[5] Strictly, towards the eigenvector, with the largest eigenvalue, to which it is nonorthogonal.

units (see Stone, 1986), as may be expected from the fact that it produces the least mean square solution. Thus, in contrast to the Hebb rule, the $w_{i,j}$ produced by the delta rule for the jth input unit will partly reflect how well the ith output unit is already predicted by other input units, just as for regression coefficients.

As for the Hebb rule, let the vector of output activations for the kth input pattern be given by $\mathbf{o}_k = \mathbf{W}\mathbf{a}_k$, and let the $\mathbf{a}_k$ be classified as "grammatical" according to their correlation with their $\mathbf{o}_k$. In contrast to a Hebbian auto associator, the weights matrix, $\mathbf{W}$, produced by a delta rule auto associator is not a covariance matrix but a matrix of regression weights. Thus, in contrast to a Hebbian autoassociator, a delta rule autoassociator does not acquire knowledge that can be interpreted in terms of PCA.

As for a Hebbian autoassociator, output activations will equal input activations only if the input activations are eigenvectors of $\mathbf{W}$. Thus, characterizing the eigenvectors of $\mathbf{W}$ would enable a characterization of what is learned by the autoassociator. If the exemplars learned are coded as linearly independent,[6] then, when learning asymptotes, any test exemplar that is a linear combination of the learning exemplars will be an eigenvector of $\mathbf{W}$ with eigenvalue equal to 1. All such test exemplars would be classified as "grammatical." Any pattern that is not a linear combination of the learning exemplars would not be an eigenvector of $\mathbf{W}$. Thus, to the extent that a test exemplar deviated from a linear combination of learning exemplars, it would be classified as "nongrammatical." If the learning exemplars are linearly dependent, it is not a priori clear how to characterize the dominant eigenvector in the general case. More will be said on this topic in the section, "Properties of Simultaneous Delta Rule Models."

The Delta Rule, the Exemplar Model, and Human Concept Formation. Recently, it has been argued that the delta rule is relevant in understanding human concept formation. The previous applications of the delta rule to experimental concept-formation paradigms are now briefly reviewed.

A number of studies have compared the delta rule with an exemplar model (described later) in accounting for human concept formation. Estes, Campbell, Hatsopoulos, and Hurwitz (1989), Gluck and Bower (1988), and Shanks (1990, Experiments 1 and 2) used a task in which subjects classified patterns of symptoms into one of two diseases, one of which was more common than the other. The delta rule network used to model this task by all three studies consisted of an input unit for each symptom connected to a single output unit coding the disease category. All three studies found that subjects' asymptotic classification data of complete symptom patterns were close to matching the normative Bayesian values, as predicted both by the network model and by the exemplar model (with no forgetting; this model is described

[6] That is, no input vector can be formed by a linear combination of the other input vectors.

later). Thus, the models did not differ in terms of the learning *outcome,* but they did differ in their trial-by-trial predictions of the learning *process,* for the stimuli employed. At asymptote, the error produced by the delta rule network on any given trial will be small, and so the weights will not fluctuate much across trials. At asymptote, the predictions of the exemplar model will be changed each trial by the acquisition of a new exemplar, regardless of the error in its prediction. Thus, due to local sequence effects, the predictions of the exemplar model may deviate more widely than the network model from Bayesian matching on any given trial.

Estes et al. (1989) found that subjects' trial-by-trial performance could be better accounted for by the network rather than exemplar model, mainly because of the deviations from Bayesian matching predicted by the exemplar model. The stability of the delta rule's predictions at asymptote differentiates it from the Hebb rule as well as from the exemplar model. It would be useful to develop stimuli that distinguished the models in terms of the learning *outcome;* this is achieved in this article by using artificial grammars (see also, Shanks, 1990, Experiment 3).

Medin and Edelson (1988) also presented results consistent with the delta rule as a model for concept formation. Their results also illustrate the importance of considering how absent features are encoded in determining the predictions of a delta rule model. Subjects classified symptom patterns into diseases, some of which were more common than others. Under some conditions, subjects showed an inverse base rate effect (i.e., incorrectly regarded the rare disease as more probable), and under other conditions, subjects showed a normative base rate effect. Medin and Edelson (1988) argued that a delta rule model could account for this pattern. Markman (1989) pointed out that the delta rule could only do so if the absence of a feature or disease is coded as -1 (and not as 0). Coding the absences of a feature as -1 means that the absence of the feature can have an effect on the activation of output units. The absence of a frequent feature can have a large negative effect on activation of output units, thereby producing an inverse base rate effect.

The relevance of these studies to modelling artificial grammar learning might be questioned because the performance of a network is as much determined by its architecture as by the learning rule used; with artificial grammar learning, the appropriate architecture would be an auto associator rather than the simple network used by Gluck and Bower (1988). Nonetheless, the relative success in the simple case examined by Gluck and Bower indicates that the delta rule should not be dismissed in examining artificial grammar learning. Furthermore, McClelland and Rumelhart (1985, 1986) argued for a delta rule autoassociator in understanding human concept formation. They found that the autoassociator could extract a central tendency or prototype from a set of patterns that were random distortions of the proto-

type (cf. Anderson, 1983), and that it could do this for several different prototypes simultaneously. Also, representations of specific exemplars could coexist in the same set of connections with knowledge of the prototype. The ability of the model to store nonorthogonal prototypes and patterns was dependent on the use of the delta rather than Hebb rule. These qualitative results are encouraging in considering modelling artificial grammar learning with a delta rule autoassociator.

Details of the Connectionist Models

Now the models specifically used in this article to model artificial grammar learning are considered. The models were all variants of an autoassociator. That is, the model attempted to predict each feature of the exemplar applied based on some set of the remaining features of that exemplar.

Criteria for assessing the performance of connectionist models have not yet crystallized in the literature. One strategy is to select parameter values for the models that optimize their performance, and to report the performance with these parameter values. Another strategy will be adopted here. Predictions will be derived from simple models; at least one of the predictions will be parameter free. The influence of key assumptions in the model will be assessed by comparing the model's predictions with and without each assumption. The models differed according to four assumptions: the learning rule used, and coding of letter features, the coding of absent features, and the use of successive versus simultaneous prediction. These assumptions are discussed in turn.

1. The Learning Rule Used. The two rules used were the Hebb rule and the delta rule. During learning, the Hebb rule is parameter free. The pattern of weights produced does not depend on a learning rate, the number of iterations through the training strings, or the sequence of string presentation.

Two learning parameters need to be considered for the delta rule: *LR*, the learning rate, and *NI*, the number of iterations through the exemplars. Before asymptotic performance, the sequence of string presentation may also be important. As long as *LR* for the delta rule is below a maximum value (see Stone, 1986, for what this is), it does not influence the final pattern of results, only how long it takes to get there. Thus, a parameter-free version of the delta rule can be produced by determining the asymptotic pattern of weight. In practice, the asymptotic weights were determined by setting *LR* at .01 or .02 and running the model for 500–1000 iterations (evidence is presented in Appendix B that the weights were indeed asymptotic under these conditions). Note that there is a peak after fewer than six iterations in the ability of the model to classify, but the *pattern* of classification does change after this point. Thus, for each delta rule model, its predictions were tested, first, with asymptotic weights, and second, with weights produced

by six iterations (in the experimental data used to assess the models, subjects were exposed to the strings six times), with the strings presented in the same order as for subjects, and with the approximately optimal learning rate for that model. "Optimal" means the learning rate that appeared to maximize classification performance, as determined by a rough "hand" exploration of *LR* space. The first type of delta rule model will be called "asymptotic," and the second type "preasymptotic."

2. The Coding of Letter Features. The material presented to the models was the same material presented to subjects; see Table 1 for a list of the acquisition and test strings. The strings to be learned were up to six letters in length, and each letter position could be filled (or not filled) with any of five different letters according to the rules of the grammar; see Figure 1 for these rules. In *single-letter* models, 30 units were used, 1 unit for each letter in each position. In *digram* models, in addition to single-letter coding, digrams were also coded. As with single letters, the same digram in different positions was coded by a different unit. Thirty-seven units were used to code the 37 allowable digrams, with 1 unit corresponding to 1 digram. Five additional units coded nonallowable digrams, 1 unit for each of the five possible digram positions. Thus, in total, 72 units were used for digram models.

Digram coding allows the model to learn interactive relations between the letters. Interactive relations were not common in the grammar used, but they did exist; for example, an R preceded by a V can only be followed by an X; but an R preceded by an X can only be followed by an R or M.

3. The Coding of Absent Features. In all models, if a feature was present, the unit coding it was given an activation of 1. If a feature was absent, the unit coding it could have an activation of 0, for one type of model, or -1, for the other type of model. In the first case, the model is sensitive to the frequency of co-occurrence of features; that is, to the frequency of (p_i+p_j), where p_i indicates the presence of the ith feature, and p_j of the jth feature. In the second case, the model can be additionally sensitive to contingency between features, that is, to the frequency of $(p_i+\sim p_j)$, or vice versa, as well as of (p_i+p_j). The two types of model will, therefore, be called co-occurrence and contingency models, respectively.

In a Hebb co-occurrence model, w_{ij} is a direct tally of the frequency of co-occurrence of features i and j (coded for by units i and j, respectively). In a delta rule co-occurrence model w_{ij} is a measure of the extent to which the occurrence of feature j uniquely predicts the occurrence of feature i. In a Hebb contingency model, w_{ij} will be decremented if feature i is present but not feature j, or vice versa. In fact, in this model, w_{ij} is a direct measure of the extent to which features i and j behave similarly, that is, of the sum of the frequencies of (p_i+p_j) and of $(\sim p_i+\sim p_j)$ minus the sum of the frequencies of $(p_i+\sim p_j)$ and of $(\sim p_i+p_j)$. In a delta rule contingency model,

again, of course, it is only the unique prediction of contingency that is important for w_{ij}.

Note that coding the absence of a feature as -1 rather than 0 implies the active coding of the absence of a feature by the subject. This is plausible when subjects are exposed to stimuli with a only small set of well-learned features. It is possible that in the grammar used (see Figure 1), the absence of a feature was as noticeable to subjects as its presence.

4. Successive Versus Simultaneous Prediction.. In successive prediction, each unit only received activation from units in previous positions. This might correspond to the case where the subject reads each stimulus from left to right. For successive models, a single, permanently active, "initial unit" was used to predict features in the first position. Thus, these models had 31 units for single-letter coding, and 73 units for digram coding. In simultaneous prediction, each unit was connected to all other units. This would correspond to the case where the subject used both previous and succeeding letters to constrain the identity of the letter in any given position.

All four assumptions were fully crossed to produce 16 different types of model. Apart from the differences discussed, all models followed the same procedure. In the learning phase, the model was exposed to each of the 20 grammatical acquisition strings used in Dienes et al. (1991) and Dulany et al. (1984). For each string, weights were changed according to the learning rule involved. One iteration through the strings was used for the Hebb rule models, six iterations for the pre-asymptotic delta rule models, and 500–1000 iterations for the asymptotic delta rule models. In the test phase, the model classified each of the 25 grammatical and 25 nongrammatical test strings used in Dienes et al. and in Dulany et al. To do this, the weights matrix of the model, $\mathbf{W}$, was used to predict the activation of each unit based on the other units (all other units, or only previous ones, depending on the model) to produce a vector of predicted activations for the kth test string, $\mathbf{p}_k$,

$$\mathbf{p}_k = \mathbf{W}\mathbf{a}_k.$$

If $\mathbf{a}_k$ is close to a "central tendency" extracted by the network, then $\mathbf{p}_k$ will closely match $\mathbf{a}_k$. Thus, the cosine, (COS) of the angle between $\mathbf{a}_k$ and $\mathbf{p}_k$ was calculated, where $\mathrm{COS} = \mathbf{a}_k \cdot \mathbf{p}_k / |\mathbf{a}_k|\ |\mathbf{p}_k|$.

In order to convert the cosine into a response probability, the procedure adopted by Estes et al. (1989), Gluck and Bower (1988), McClelland and Elman (1986), and McClelland and Rumelhart (1986) was employed. The probability of responding "grammatical" to a string was taken to be the sigmoid function,

$$p(\text{"}g\text{"}) = 1/(1 + e^{-k\cos - T})$$

where k is a scaling parameter and T is a threshold; k gives a degree of freedom in adjusting predicted to actual, overall response probabilities. T

is adjusted to give equal numbers of "grammatical" and "nongrammatical" responses (a program iteratively tries different T values, calculates the average $p(\text{"}g\text{"})$ for all test strings, and then adjusts T slightly upwards if the average $p(\text{"}g\text{"})$ is less than .50, and adjusts T slightly downwards if the average $p(\text{"}g\text{"})$ is more than .50. It is assumed that subjects would do this on-line, adjusting their thresholds according to how many "grammatical" responses they have given so far).

From the $p(\text{"}g\text{"})$ for each string, the proportion of strings expected to be (1) classified correctly overall (Pc), (2) classified correctly twice in a row (CC), (3) classified correctly once and in error once (CE), and (4) classified in error twice in a row (EE) over two classification blocks could be calculated. If k could be adjusted to give the same pattern of values for Pc, CC, CE, and EE as were obtained in Dienes et al. (1991) and Dulany et al. (1984), this would provide an existence proof that the models could match overall characteristics of the experimental data given appropriate parameter tweaking. If, for a given model, there was no value of k for which the experimental patterns of values could be obtained, then the model would be clearly inadequate.

From the cosine for each string, a rank order of difficulty for grammatical and nongrammatical strings was constructed for each model. For the Hebb rule and asymptotic delta rule models, these rank orderings were parameter-free predictions of the models. For pre-asymptotic delta rule models, the parameters were not determined by their influence on the rank orderings. As long as the rank orderings are different for different models, the correspondence between experimentally obtained and predicted rank orderings can be used to test the different models competitively.

Memory-Array Models

Introduction. One key point of debate in the implicit learning literature has been whether implicit knowledge is best represented in an abstract way (e.g., Reber, 1989; Reber & Allen, 1978) or in terms of the storage and deployment of exemplars (e.g., Brooks, 1978; Ericsson & Simon, 1984, p. 114). Brooks (1978) showed that subjects *can* use analogy to stored exemplars to classify at above-chance levels. Reber and Allen (1978) argued that this was not the normal strategy of subjects. They presented subjects either with a paired associate learning task (each string was paired with a city name) or a task that simply involved observing strings. They observed several differences in the way subjects subsequently categorized grammatical and nongrammatical strings; the results do suggest two different strategies, but do not rule out an exemplar model of both.

McAndrews and Moscovitch (1985) sought to determine whether rule-based or exemplar-based information was a more important determinant of

classification performance in artificial grammar learning. Grammaticality and similarity to studied strings were manipulated independently. Similarity was measured (inversely) by the smallest number of differences in letter positions to any studied string. Grammaticality and similarity were found to account for a roughly equal amount of variance in classification performance. McAndrews and Moscovitch concluded that there was evidence for the abstraction of rule-based information. However, an exemplar model could account for the effects of both similarity and grammaticality. The grammaticality effect may arise because each letter position of a grammatical item is likely to be the same as the letter position of a large subset of stored exemplars (though nonidentical subsets for different letter positions); on the other hand, some letter positions of nongrammatical items may be different to the letter positions of any stored exemplars. The issue is best resolved by actually running simulations of different models. Whether exemplar models like those of Estes (1986) or Hintzman (1986) can account for the pattern of classification performance under observational learning conditions is an open question.

Details of the Memory-Array Models. Three types of memory-array model were considered: The exemplar model of Estes (1986; also, Medin & Schaffer, 1978), the feature-array model of Estes (1986), and the multiple trace model of Hintzman (1986). These are discussed in turn.

(a) The Exemplar Model of Estes (1986) and Medin and Schaffer (1978). According to the exemplar model, exemplar information is stored as an array of feature values. In the simplest model, all acquisition exemplars are stored perfectly. The first step in categorizing a test exemplar is to determine its similarity to each of the acquisition exemplars. This computation is done by entering a parameter, s_i ($0 \leq s_i < 1$), for each feature i where the test and acquisition exemplars have different values, entering a 1 for each feature with common values, and taking the product. Thus, if the test and acquisition exemplars differ on n features, then the computed similarity is

$$\prod_{i=1}^{n} s_i$$

or s^n, if $s_i = s$, for all n. So, s_i can be regarded as reflecting the salience of the contrast between different values of the ith feature: The lower s_i is, the greater the salience. The probability of categorizing a test exemplar as "grammatical" is a function of the sum, A, of its similarities to all acquisition exemplars:

$$p(\text{“}g\text{”}) = 1/(1 + e^{-kA+T})$$

as for the connectionist models.

The exemplar model could be implemented by a connectionist model with an input layer and two layers of hidden units in the following way: The string to be classified is applied to the input layer, where each unit represents one feature (an activation of +1 for the presence of a feature and −1 for its absence). In the next layer up, there is one hidden unit for each stored exemplar. The weights to each hidden unit represent the feature values of that exemplar (+1 or −1). These hidden units would be pi units, multiplying together the weighted input from all its connections (a weighted input of −1 would be converted to a value s) to produce their output activations. With only two categories (grammatical or not), the next layer would contain only one hidden unit (an ordinary sigma unit) which simply summed the activations of all the other hidden units. The activation of this unit determines the probability of classifying a given input string as "grammatical."

Medin and his colleagues provided considerable evidence that the multiplicative relationship used in combining similarities for (experimenter-defined) features is important in accounting for classification performance across a range of tasks. This multiplicative relationship allows the model to be sensitive to correlations between features, and not just their independent effects. Note that this aspect of the exemplar model distinguishes it from prototype theories and also networks using the Hebb or delta rules, which employ an additive combination of information. That is, the prototype and Hebb and delta network models can only solve *linearly separable* classification tasks: The category to which an exemplar belongs must be predictable from a linear combination of the features values used to encode the exemplar.

Medin and Schwanenflugal (1981) showed that subjects learned nonlinearly separable classification tasks just as easily as linearly separable ones. Furthermore, Medin, Altom, Edelson, and Freko (1982) found that even when a classification task could be solved in a linearly separable way, subjects still preferred test exemplars that preserved correlations. Kemler-Nelson (1984; see also Kemler-Nelson, 1988; Ward & Scott, 1987) showed that incidental but not intentional learners found a nonlinearly separable task easier than a linearly separable one. This is interesting because of the incidental conditions under which subjects typically learn artificial grammars. Although these data are consistent with an exemplar model of concept formation, they do not rule out Hebb or delta rule models if these models include an initial nonlinear process that presents combinations of experimenter-defined features to the Hebb and delta rule networks (e.g., consider the digram coding employed in the connectionist models earlier).

Four exemplar models were considered: ex1, ex2, ex3, and ex4. For all of them, the features used to code each exemplar were the 30 letter-position features used for the single-letter connectionist models. The four models differed according to how s_i changed with letter position. For ex1, $s_i = s = .1$ for all i (the exact value of s made little difference to the model over the

range .001–.5). This model assumed that all letter positions were equally salient to the subjects. For ex2, s_i increased linearly with letter position, from .1 for Letter Position 1 and .6 for Letter Position 6. This model assumed that the initial letters were most salient and the final letters least salient. For ex3, s_i varied quadratically with Letter Positions 1 to 6, with a minimum of .1 for Letter Positions 1 and 6, and a maximum of .6 for Letter Positions 3 and 4. This model assumed that Letter Positions 1 and 6 were most salient to subjects. And for ex4, s_i varied quadratically with the beginning and end of the test exemplar, regardless of the absolute letter position. This model assumed that the beginning and end letters (at whatever absolute letter position) were most salient.

(b) The Feature-Array Model of Estes (1986). The feature probability array model of Estes (1986) uses the same memory array as the exemplar model. Categorization relies on the "perceived frequencies," f_i, of each feature value i contained in the test exemplar over all acquisition exemplars. Hence, f_i is incremented by 1 for each acquisition exemplar containing feature value i, and by s, $0 \leq s < 1$, for each exemplar not containing feature value i. The probability of classifying a test exemplar is a function of

$$L = \pi_i f_i$$

where i is over all the feature values characterizing the test exemplar:

$$p(\text{"}g\text{"}) = 1/(1 + e^{-kL+T}),$$

as before.

The feature-array model could be represented by a network with an input layer whose units code the features of the test string (with activations of $+1$ and -1 for the presence and absence of a feature, respectively), and a single pi hidden unit, whose weights are the f_i. The hidden unit would multiply together only the positive weighted inputs.

The features used to code each exemplar were the 30 letter-position features used for the single-letter connectionist models and the exemplar models.

Estes (1986) showed that when each cue of a pattern independently predicts category membership with a given probability, the feature- and exemplar-array models fare equally well in accounting for subject performance. However, as for the connectionist models, the feature-array model cannot predict learning when it is only combinations of the features encoded by the model that predict category membership; in this situation, the exemplar-array model fares better (Estes, 1986).

(c) The Multiple Trace Model of Hintzman (1986). The MINERVA 2 model of Hintzman (1986) was an attempt to show how abstract concepts

could be acquired and represented in a system that stored only episodic traces. Briefly, when a probe is presented to primary memory, it activates all traces in secondary memory according to how similar they are to the probe. This results in an echo with *intensity* and *content* returning to primary memory. Echo intensity depends on the total amount of secondary memory activation triggered by the probe, and forms the basis of judgements of familiarity. Echo content depends on which particular features in secondary memory are strongly activated.

Traces are stored as feature lists where 1 codes the presence of a feature and -1 its absence. If P_j represents the value of the jth feature of the probe, and T_{ij} represents the value of the jth feature of trace i, then the similarity of the probe to trace i is given by the correlation between the two vectors, that is

$$S_i = (1/N) \sum_{j=1}^{N} P_j T_{ij}$$

where N is the number of relevent features. So far, similarities have been combined additively across different features; the degree of activation of trace i is

$$A_i = S_i^3.$$

The cubic function increases the signal to noise ratio in the echo (see Hintzman, 1986). Raising S_i to a power greater than 1 also introduces some multiplicative terms between similarities from different features, as in Medin and Schaffer's (1978) model.

Intensity is found by summing activation over all m traces

$$I = \sum_{i=1}^{m} A_i.$$

The activation of each feature in the echo (i.e., echo content) is

$$C_j = \sum_{i=1}^{m} A_i T_{ij}.$$

Hintzman (1990) indicated how MINERVA 2 could be implemented as a two-layer connectionist model, with one input-output layer and a layer of hidden units. The features of an input, for example, a test string, are represented by the activation of the input-output units. Each hidden unit represents one trace, with the weights representing feature values. The activation of each hidden unit is a cubic function of its net input. This activation passes back down to the input-output layer, summating the activation from all hidden units, to produce the echo.

Hintzman (1986) showed that MINERVA 2 could simulate a number of results from the concept-formation literature: Better classification of proto-

types than old exemplars with a delayed test (Posner & Keele, 1970), effects on classification of category size and of the extent of the distortion used to generate exemplars from prototypes (Homa & Vosburgh, 1976), and the effect of within-category similarity among exemplars (Elio & Anderson, 1981).

For this article, MINERVA 2 was used to classify exemplars based on either echo intensity or echo content. In both cases, it was assumed that all $m = 20$ acquisition exemplars had been stored. Each of the 50 test exemplars were used as probes. When echo intensity was used to classification was given by

$$p(\text{“}g\text{”}) = 1/(1 + e^{-kI+T})$$

as with the connectionist models. In this equation, k could be regarded as referring to the number of traces stored of each exemplar. Because I is simply the sum of the activations of each trace, increasing the numbers of each trace by a factor k would increase I by a factor k as well. When echo content was used, the correlation C between the probe and the echo content was calculated, and classification was given by

$$p(\text{“}g\text{”}) = 1/(1 + e^{-kC+T}).$$

In this case, increasing the numbers of each trace would leave the pattern of the content unchanged, and hence k could not refer to the numbers of each trace.

The basis of classification (intensity vs. content) was crossed with type of feature coding (single letters alone vs. single letters and digrams to produce four versions of the model.

For all memory-array models—the exemplar array, feature array, and MINERVA 2 models—k was adjusted so as to produce Pc, CC, CE, and EE values as close as possible to experimental values. The rank ordering of exemplar difficulty depended on the parameter s_i for the exemplar- and feature-array models, but was a parameter-free prediction of the MINERVA 2 models.

The reader is referred to Appendix A for data indicating the extent to which the different connectionist and memory-array models made different predictions about the rank order of exemplar difficulty. In essence, the Hebb rule, delta rule, and Estes (1986) models made substantially different predictions to each other, and the co-occurrence–contingency and successive–simultaneous assumptions also introduced substantial variation in the predictions. On the other hand, all the MINERVA 2 models made similar predictions to the Hebb contingency models, and the single-letter models made similar predictions to the digram models.

EVALUATION OF THE MODELS

Models were evaluated in terms of the Pc, CC, CE, and Ee values they could produce, and also in terms of the rank ordering of exemplar difficulty

that they predicted. The use of rank order of exemplar difficulty allows a direct test of the different learning processes in the models independent of the particular probability function employed to produce Pc values. This use of nonparametric rather than parametric tests allows more general conclusions about the nature of the models. A parametric test, like least squares, would test the learning process and probability function as a whole for each model, and would thus be less informative about why a particular model worked and others did not. (Was it the learning process in the model or the interaction of the learning process with the probability function? Can we be sure that our parameter search was complete? This can be a difficult question for some connectionist models. Might the model produce any behavior we wish just by tweaking parameters?) Nonetheless, to indicate that a parameterization is possible, the most successful model according to the nonparametric tests will be fitted against the empirical data using least squares.

The k parameter used to scale response probabilities in all the models was adjusted so as to maximize Pc with the constraint that $EE - EC \approx .05$. In general, increasing k would increase both Pc and EE[7]. If the maximum Pc so obtained for a model exceeded empirically obtained values, then, with suitable parameter tweaking, the models' values could be made to match experimental values (by reducing k and/or adding noise to deteriorate performance). On the other hand, if the maximum Pc was below empirically obtained values, then no parameter tweaking could rescue the model. As a guide for deciding a reasonable lower limit for Pc, the confidence limits were calculated for the best group in Dienes et al. (1991; these were the single-task subjects of Experiment 2). Their mean Pc was .69, and the lower limit of their 95% confidence limit was .65. Thus, any model that produced a maximum Pc below .65 would be unsatisfactory as a model of artificial grammar learning.

Pre-asymptotic delta rule models were tested against the RANK1 rank ordering; all other models were tested against the TOTAL rank ordering. The reason for testing the different models against different data is that the pre-asymptotic models were sensitive to the order of presentation of the exemplars, and the order used for the pre-asymptotic models was the same as that used by the subjects in Dienes et al. (1991), but not the same as that used by Dulany et al. (1984). Thus, only the RANK1 data was appropriate for the pre-asymptotic models. On the other hand, the other models were not sensitive to presentation order, and the TOTAL rather than RANK1 data gives a better estimate of the subjects' rank order of exemplar difficulty independent of presentation order.

[7] Thus, these models might seem to predict that Pc and EE should be correlated. Unfortunately, increasing LR in the delta rule models can increase Pc and decrease EE, before learning asymptotes, thereby introducing a negative correlation. However, there should be a positive correlation when subjects' learning asymptotes; this would be an interesting prediction for future studies to test.

TABLE 3
Classification by Hebbian Models

	Co-occurrence				Contingency			
	Single		Digram		Single		Diagram	
	Succ	Sim	Succ	Sim	Succ	Sim	Succ	Sim
Pc	.70	.73	.67	.83	.56	.65	.55	.64
CC	.57	.61	.54	.76	.37	.51	.34	.47
CE	.12	.11	.13	.06	.19	.14	.21	.16
EE	.18	.16	.19	.11	.25	.21	.25	.21

Note. Pc is the proportion of strings correctly classified; CC is the proportion correctly classified twice in a row; CE is the proportion correctly classified once correctly and once in error; and EE is the proportion classified in error twice in a row.

TABLE 4
Rank Correlations Between String Difficulty for Subjects and Hebbian Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Co-occurrence				
Grammatical	−.24	−.26	−.19	−.20
Nongrammatical	.48	.52	.44	.54
Contingency				
Grammatical	−.08	−.29	−.06	−.26
Nongrammatical	−.14	.12	.20	.08

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

The Hebbian models are considered first, then the delta rule models, and finally the memory-array models.

Hebbian Models

Table 3 shows the Pc, CC, CE, and EE values for the different Hebbian models. The Pc values produced by the contingency Hebbian models (which code the absence of a feature as −1 rather than 0) are too low, and so these models are not adequate models of artificial grammar learning. The co-occurrence models pass this initial test.

Table 4 shows the correlations between the rank ordering of string difficulty predicted by the Hebbian models and TOTAL. The co-occurrence models could significantly predict the rank order of nongrammatical string difficulty. However, neither they nor the contingency models could predict the empirical rank order of grammatical string difficulty. Thus, none of the Hebbian models are adequate models of artificial grammar learning.

Delta Rule Models

Table 5 shows the Pc, CC, CE, and EE values produced by the asymptotic delta rule models, and Table 6 shows the values for the pre-asymptotic delta

TABLE 5
Classification by Asymptotic Delta Rule Models

	Co-occurrence				Contingency			
	Single		Digram		Single		Diagram	
	Succ	Sim	Succ	Sim	Succ	Sim	Succ	Sim
Pc	.73	.75	.72	.78	.73	.70	.67	.78
CC	.62	.66	.60	.69	.63	.58	.53	.69
CE	.11	.10	.12	.09	.11	.13	.14	.09
EE	.16	.15	.16	.14	.16	.18	.19	.13

Note. Pc is the proportion of strings correctly classified; CC is the proportion correctly classified twice in a row; CE is the proportion correctly classified once correctly and once in error; and EE is the proportion classified in error twice in a row.

TABLE 6
Classification by Pre-asymptotic Delta Rule Models

	Co-occurrence				Contingency			
	Single		Digram		Single		Diagram	
	Succ	Sim	Succ	Sim	Succ	Sim	Succ	Sim
Pc	.72	.87	.72	.87	.72	.79	.71	.81
CC	.61	.83	.61	.83	.61	.72	.59	.73
CE	.12	.04	.11	.04	.11	.08	.12	.07
EE	.16	.09	.17	.09	.16	.13	.17	.12

Note. Pc is the proportion of strings correctly classified; CC is the proportion correctly classified twice in a row; CE is the proportion correctly classified once correctly and once in error; and EE is the proportion classified in error twice in a row.

rule models. All the delta rule models could produce reasonable values for Pc. Interestingly, in many cases, as the ability of the delta rule models to complete the acquisition strings improved (from pre-asymptotic to asymptotic; see Appendix B), their ability to generalize to new strings deteriorated. Brooks (as reported in McAndrews & Moscovitch, 1985) suggested an exemplar model interpretation of this pattern of performance; the results here show that the pattern is not in itself indicative of an exemplar model.

Tables 7 and 8 show the correlations between the predicted and actual rank order of string difficulty for the asymptotic and pre-asymptotic models, respectively. The actual rank ordering was TOTAL for the asymptotic models and RANK1 for the pre-asymptotic models.

The successive–simultaneous distinction (i.e., whether features were predicted only by previous features in the string or by all features in the string) was of most importance in distinguishing the different delta rule models: The successive models were unable to predict the rank order of grammatical

TABLE 7
Rank Correlations for String Difficulty Between Subjects and Asymptotic Delta Rule Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Co-occurrence				
Grammatical	.02	.57	−.02	.61
Nongrammatical	.34	.40	.32	.53
Contingency				
Grammatical	.01	.56	−.17	.69
Nongrammatical	.16	.52	.12	.38

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

TABLE 8
Rank Correlations for String Difficulty Between Subjects and Pre-asymptotic Delta Rule Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Co-occurrence				
Grammatical	−.15	.36	−.12	.42
Nongrammatical	.50	.52	.54	.53
Contingency				
Grammatical	−.14	.57	−.13	.44
Nongrammatical	.06	.52	.04	.36

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

strings; the simultaneous models could significantly predict the rank order of both grammatical and nongrammatical strings. Indeed, for grammatical strings, the difference in the size of correlation between corresponding models differing only with respect to the successive versus simultaneous distinction was significant (all $ps < .01$) for all eight comparisons (the eight comparisons are obtained by crossing co-occurrence versus contingency, single versus digram, and asymptotic versus pre-asymptotic) using William's test for nonindependent correlations (see Howell, 1987, p. 243).

The single-letter–digram and co-occurrence–contingency distinctions did not influence the ability of the autoassociator to predict the rank order of string difficulty. The difference in the size of correlation between corresponding models differing only with respect to the digram versus single-letter distinction was not significant (all $ps > .10$) for any of the 16 comparisons using William's test. The difference in the size of correlation between corresponding models differing only with respect to the co-occurrence versus contingency distinction was significant ($p < .05$) for only 2 of the 16 com-

TABLE 9
Classification by Memory-Array Models

Estes (1986):	ex1	ex2	ex3	ex4	fre
Pc	.71	.78	.56	.63	.56
CC	.60	.69	.33	.48	.36
CE	.13	.09	.22	.16	.19
EE	.17	.15	.22	.22	.25
MINERVA 2:	DI	DC	SI	SC	
Pc	.63	.73	.62	.72	
CC	.47	.62	.45	.61	
CE	.16	.11	.17	.11	
EE	.21	.16	.21	.16	

Note. Pc is the proportion of strings correctly classified; CC is the proportion correctly classified twice in a row; CE is the proportion correctly classified once correctly and once in error; and EE is the proportion classified in error twice in a row. For the MINERVA 2 models: I=categorization by intensity; C=categorization by content; S=single-letter coding; D=digram coding.

parisons (none of the 8 comparisons for grammatical strings was significant, $ps > .10$).[8]

Memory-Array Models

Table 9 shows the Pc, CC, CE, and EE values produced by the memory-array models. The ex1 and ex2 exemplar models, and the MINERVA 2 models that classified by content rather than intensity, showed adequate Pc values. The other models were not adequate in this respect.

Hintzman (1986) suggested that the performance of MINERVA 2 could be improved in some situations if the echo returning from a probe is itself fed back to secondary memory to produce a new echo; this procedure can be repeated for a number of iterations. When this procedure was followed, renormalizing the content vector to a standard length after each iteration, the Pc values for the MINERVA 2 models actually deteriorated. After one iterations, the Pc for the intensity models fell to chance. After four iterations, the performance of the content models also fell close to chance. (A similar procedure can be followed for the connectionist models: **W** can be applied a number of times to the resulting **p**, renormalizing **p** to a standard length each time, until a stable state is reached. This essentially amounts to comparing each **a** to its nearest eigenvector. As for MINERVA 2, this pro-

[8] The single simultaneous delta rule model was run again (asymptotically) with the absence of a feature coded by −0.5, rather than by −1 (contingency model) or 0 (co-occurrence model). This may be a more plausible coding assumption (thanks to Arthur Reber for making this suggestion). This model correlated highly with both the contingency model (.78 and .97 for grammatical and nongrammatical strings, respectively) and the co-occurrence model (.74 and .89, respectively). It also correlated well with the TOTAL data (.43 and .55, respectively).

TABLE 10
Memory-Array Models

Estes (1986):	ex1	ex2	ex3	ex4	fre
Grammatical	−.08	−.14	−.04	−.05	−.16
Nongrammatical	.41	.43	.18	.23	.08
MINERVA 2:	DI	DC	SI	SC	
Grammatical	−.22	−.12	−.30	−.08	
Nongrammatical	.03	.17	.12	.27	

Note. For the MINERVA 2 models: I=categorization by intensity; C=categorization by content; S=single-letter coding; D=digram coding. $r(\text{crit})_{.05}=.34$, one-tailed, or .40, two-tailed.

cedure decreases Pc values for all the models except the Hebb co-occurrence models, which are only marginally improved.)

Table 10 shows the correlations between the predicted and actual rank order of string difficulty for the memory-array models. Ex1 and ex2 could significantly predict the rank order of nongrammatical string difficulty, but no memory-array model could predict the rank order of grammatical string difficulty.

PROPERTIES OF SIMULTANEOUS DELTA RULE MODELS

The only class of model that could produce adequate Pc values and predict the rank order of both grammatical and nongrammatical strings was the simultaneous delta rule model. The fact that there were other models inadequate in each of these respects indicates that the achievement is not trivial. It is worth noting that no other model could predict the rank ordering of grammatical string difficulty.

In order to indicate how closely the simultaneous delta rule models could be made to fit the data with a parametric test, the probability of responding "grammatical" for each of the 50 exemplars was determined for different k values for the digram contingency model. The proportion of "grammatical" responses actually made to each exemplar was regressed against the model's values. For a k of 8, the Pearson correlation was .73, and the error variance was 0.013. The goodness of the fit is indicated by the intercept ($a=0.07$) being nonsignificantly different from 0 ($p>.10$) and the slope ($b=0.81$, $SE_e=0.11$) being nonsignificantly different from 1 ($p>.05$). The properties of simultaneous delta rule models are now explored further. Initially, the dependence of their success on the arbitrary type of coding used is explored. Then, an attempt is made to characterize the type of knowledge acquired by the simultaneous delta rule autoassociators in as high level a way as possible.

The models all used a position-specific form of coding: One unit represented each letter position. Thus, the same letter in different positions is as different to the model as different letters. Surely, this does not accurately represent the state of affairs in a subject's head: It seems likely that a subject might not clearly differentiate a T in the fourth position or a T in the fifth position. Furthermore, the models assumed that each letter position was accurately encoded by the subject on each trial, which may not be entirely correct. To address these problems, and thus to determine the generality of the success of the simultaneous delta rule models, three additional types of coding assumptions were tried. First, it was assumed that not all letters were successfully encoded by the subject. Specifically, for the single-letter models, it was assumed that there was a 20% probability, independently for each position, that the letter would not "be noticed," and thus would be coded as absent. For any particular run of the pre-asymptotic models, the correlation of rank order of string difficulty between the model and subject data could be substantially reduced with this noisy rather than with accurate coding. Thus, an average was taken of the ranks of 10 runs of both the co-occurrence and contingency pre-asymptotic models. For the co-occurrence model, the noisy coding reduced classification performance from .87 to .70. The average rank ordering still correlated with subject data: The r_S was .34 for grammatical strings and .51 for nongrammatical strings. For the contingency model, the noisy coding reduced classification performance from .79 to .70. The average rank ordering also still correlated with subject data: the r_S was .40 for grammatical strings and .41 for nongrammatical strings. In short, the models, even pre-asymptotically, were robust to random noise in the coding.

A second coding scheme used coarse coding for position for each letter. Each letter was represented by a set of units. A letter in a given position was represented by a pattern of activation of this set of units. The same letter in another position was represented by a different pattern of activation, but the closer the positions were, the more the patterns overlapped. Thus, this form of representation treats the same letter in neighboring positions as very similar, and the same letter in positions far away as less similar. To implement specifically this scheme, each letter was represented by 11 units. There were 55 units in total, 11 for each of the five letters. Consider the set of units representing M. Their default activation was 0. If there was an M in the first position, then the Units 1 to 6 received an activation of 1. If there was an M in the second position, Units 2 to 7 received an activation of 1, and so on. If there was an M in more than one position, the overall activation was simply the sum of the activations of the units for the M in each of the positions alone. This allows a unique coarse-coded representation of all the strings. For the asymptotic model, classification performance was .65. The correlation of string difficulty between model and subject data was .46

for grammatical strings and .46 for nongrammatical strings. For the pre-asymptotic model, classification performance was .68. The correlation of string difficulty between model and subject data was .33 for grammatical strings and .59 for nongrammatical strings. In short, with position coarse coded, the simultaneous delta rule produced rank orders of string difficulty similar to those of subjects. Very similar results were obtained if each position were represented by the activation of overlapping sets of two, three, four or five units instead of six.

A third coding scheme did away with position-specific coding altogether in favor of local context sensitivity.[9] Each unit represented three letters, for example, one unit represented $_{M}V_{T}$ (a V preceded by an M and followed by a T). Beginning and end markers were also used. Thus, the string "MTV" would be represented by activating units $_{B}M_{T}$, $_{M}T_{V}$, and $_{T}V_{E}$. This coding scheme allowed a unique representation of almost all the strings (there were two exceptions: MTTTTV and MTTTV were not distinguished, and VXRRRR and VXRRR were not distinguished). There were 82 different trigrams in all the grammatical and nongrammatical strings employed. Thus, 82 units were used in the model. The presence of a trigram was represented by setting the activation of the corresponding unit to 1.

With co-occurrence coding, the asymptotic model produced a classification performance of .75. The correlation of rank order of string difficulty between model and subjects was .29 for grammatical strings and .24 for nongrammatical strings. This is not very impressive; however, with the pre-asymptotic model—trained with the same sequence as subjects—classification performance was .74, and the correlations were .64 for grammatical strings and .35 for nongrammatical strings. With contingency coding, the asymptotic model produced a classification performance of .65. The correlations were .36 for grammatical strings and .01 for nongrammatical strings. The pre-asymptotic model produced a classification performance of .65. The correlations were .37 for grammatical strings and .16 for nongrammatical strings. In short, a model using context-sensitive coding produced similar results as subjects with pre-asymptotic co-occurrence coding, but not contingency coding.

In summary, implementation of three different coding assumptions confirmed the usefulness of the simultaneous delta rule. Specifically, the simultaneous delta rule could produce similar results as subjects with noisy as well as accurate coding of each letter position, with coarse coding of position, and with context-sensitive coding of each letter.

This section now attempts to derive some symbolic rules that describe the behavior of the simultaneous delta rule models. This is important for three reasons. First, if the models can be described in a reasonably high-level way,

[9] Thanks to James McClelland for suggesting this coding scheme.

a better understanding is obtained of the behavior of the models, and there is a greater chance that further predictions can be clearly derived from the models. Second, the more general the class of models that empirical results speak to, the more informative the results are (Broadbent, 1980). Thus, characterizing the knowledge acquired by simultaneous delta rule models in as general a way as possible would allow the results obtained here to speak to the whole class of models that satisfy the characterization. And third, the relationship of the characterization of the model's knowledge to the model might correspond to the relationship between grammatical rules and the subject: The subject or the model obeys the rules, but does not represent them symbolically. This would be consistent with the failure of the subject to describe such rules in free recall. The question of what—if any—characterization or higher level description can be given to the knowledge acquired by the simultaneous delta rule models will be addressed here with respect to their predictions of rank order of exemplar difficulty.

All the simultaneous delta rule models that used position-specific coding (i.e., not the trigram models) made very similar predictions for rank order of string difficulty for the test strings used, so characterizing the knowledge of one model will approximately do so for the others. Consider an asymptotic model trained on linearly independent strings. All the training strings will form eigenvectors of the **W** for the model and so the training strings will all be considered perfectly grammatical by the model. In fact, any linear combination of the training strings will form an eigenvector of **W**, and so will also be considered as perfectly grammatical.

For illustration, consider the acquisition strings used in this article, as shown in Table 1. Consider a single-letter model trained on the first 13 strings, which are linearly independent. If String 3 (MTV) is subtracted from String 4 (MTVRX), the stem ...RX remains (where a "." indicates an empty position) and can be added to any three-letter string. If it is added to String 11 (VXM), for example, the linear combination VXMRX is obtained (which is nongrammatical by the finite state grammar).

In general, if any two strings are the same in k positions and different in $(n-k)$ positions, where n is the maximum length of the strings, and a third string is similar to any one of the two strings in each of the $(n-k)$ positions, then a string formed from the third one by substituting the feature of the dissimilar for the similar original two strings in all $(n-k)$ positions will also be accepted as "grammatical" by the model. Call this the principle of combination.

The acquisition strings were linearly dependent for single-letter models, but were linearly independent for digram models. The effect of using digram coding with the above principle of combination is that no linear combination of acquisition strings will have illegitimate digrams, and so the correspondence between the set of strings judged "grammatical" that ac-

tually are grammatical, will, in general, be greater for the digram than for the single-letter models. This difference between digram and single-letter models was not well exploited by the test strings used, as almost all of the nongrammatical strings incorporated illegitimate digrams. Considering the strings shown in Table 1, the preceding principle of combination produces the following "rules" that are followed by the digram models:

..TTVT = ..TVT (1)
...VRX = ...VT (2)
MTV.. = VXV.. (3)
....V = ...VT (5)
.TV = .TTVT (6)
.XM = .XRR (7)
....X = ...XV (8)

and so on. For example, Rule (1) is produced by taking the difference between Strings 15 (VXTTVT) and 17 (VXTVT). Any string fitting the mold on the left or right side, where "." means any letter, can be transformed to the mold on the other side (keeping the "." letters constant), and will still be regarded as "grammatical" by the digram models. All these rules operating on grammatical strings will produce only grammatical strings.

In general, a digram model will accept as "grammatical" nongrammatical strings if a finite state grammar is used with the following necessary and sufficient properties: (1) the same letter in the same position can occur through different routes; (2) the letter can be followed by at least one letter in common through the different routes; and (3) the letter can be followed by at least one different letter through the different routes. With a suitable set of grammatical training strings, the digram model will accept as "grammatical" a string in which the common letter has been swapped for the different letter. The finite state grammar used here does not have these properties. Thus, all the rules abstracted by the model will be representative of the grammar. However, a model trained on only a subset of strings will not, in general, have a complete representation of the grammar. Interestingly, this is the state of affairs claimed by Reber (e.g., 1989) for subjects: They have representative but incomplete knowledge of the grammar. However, if the delta rule model is a valid model of subjects' knowledge, then subjects will not abstract representative knowledge from a set of grammatical strings if an appropriate finite state grammar is used.

A standard procedure given in elementary linear algebra books (see, e.g., Venit & Bishop, 1985) was used to determine the linear dependence of the test strings on the acquisition strings. All 70 strings were coded as column vectors in a 72×70 matrix. The row-reduced echelon form of this matrix revealed that 13 of the 25 grammatical test strings were linear combinations of the acquisition strings, but, of course, none of the nongrammatical test

strings were linear combinations. These 13 grammatical test strings were, of course, the 13 best-classified strings by the digram models. Eleven of these 13 were among the 13 best-classified grammatical strings by the single-letter models, which approximate the digram models with this data set. Were these 13 strings anything special as far as subjects were concerned? In fact, 11 of these 13 were among the 13 best-classified grammatical strings in the TOTAL data. This is what would be predicted by any model that consisted of all the rules that could be produced by combining the acquisition exemplars according to the principle of combination described before.

However, the delta rule models do more than just embody the rules: They can also classify, to varying degrees, grammatical strings that do not fit the rules. Consider the 12 grammatical test strings that were not linear combinations of the acquisition strings, and therefore did not follow from these rules. The rules in themselves do not speak to how these strings should be classified, but the delta rule models still do. Does the rank ordering of the delta rule models for these strings match that of subjects? The correlations between TOTAL and the rank order predicted by the co-occurrence and contingency digram models were .35 and .78, respectively (.50 is needed for significance at the 5% level), indicating some degree of match. Also, note the significant correlations between TOTAL and the predictions of the digram models for the nongrammatical test strings. Thus, the digram models provide a measure of the *extent* to which the rules are broken that matches the measure of subjects.

To summarize, the digram simultaneous delta rule model trained on linearly independent acquisition strings can be regarded as embodying (1) a set of incomplete but (for this grammar) representative rules, and also (2) a measure of deviation from the rules. If the measure of deviation can itself be characterized, then the empirical results for rank order of string difficulty would support all models of this class. Future research could usefully characterize the measure of deviation and also the knowledge of models trained on linearly dependent acquisition strings.

CONCLUSION

This article tested a range of connectionist and exemplar models for their ability to account for artificial grammar learning. The criteria used were the ability to produce adequate levels of Pc and of (EE – EC), and to predict the rank order of both grammatical and nongrammatical strings. Only one class of model—the simultaneous delta rule model—could satisfy all the criteria. In fact, the simultaneous delta rule model was the only model that could predict the rank order of grammatical string difficulty. It was shown that the classification knowledge of the digram versions could be regarded as embodying representative but incomplete rules of the finite state grammar.

Although successful by the criteria of this article, it would not be difficult to falsify simple models like the simultaneous delta rule autoassociator. For example, without the addition of further processes, it would not be able to account for the transfer of knowledge from one letter set to another (Mathews et al., 1989; Reber, 1969). Also, certain aspects of the encoding characteristics appear arbitrary: The same letter in different positions is as different to the model as different letters in different positions (this problem was partially overcome by coarse-coding position and also by using trigrams independent of position). Finally, why should a subject start off with a unitary representation of each digram, as assumed by the digram models?

A future direction for modelling artificial grammar learning may be to introduce a procedure for extracting features by creating useful higher level units out of lower level ones. This could be achieved by, for example, competitive learning algorithms with hidden units (Rumelhart & Zipser, 1986) or the algorithm used by Wolff (1975, 1977, 1980) to segment prose into meaningful units. A particularly useful architecture employing hidden units might be the recursive network of Elman (1990), used by Cleeremans and McClelland (1991) to model the sequential learning of a noisy finite state grammar in a reaction time paradigm. The use of hidden units might allow the creation of unitary representations of abstract structures in the grammar not tied to particular letters, a representation of the fact that the same letters in different positions are the same letters, and also the creation of unitary representations of digrams in a natural way. Perhaps the type of model used in this article is best regarded as a second stage operating on the emerging units created by a first-feature extraction stage. Servan-Schreiber and Anderson (1990) proposed an ACT* account of feature extraction as a sufficient model of artificial grammar learning. However, it remains an open question as to whether their model could predict, parameter free, the rank order of string difficulty as well as the simultaneous delta rule models.

Another issue deserves comment. The models used in this article were particularly geared to artificial grammar learning. Future research needs to address the applicability of delta rule autoassociator to other learning paradigms.

REFERENCES

Anderson, J.A. (1983). Cognitive and psychological computation with neural models. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-13,* 799–815.

Broadbent, D.E. (1980). The minimization of models. In A.J. Chapman & D.M. Jones (Eds.), *Models of man.* Leicester, England: British Psychological Society.

Brooks, L. (1978). Nonanalytic concept formation and memory for instances. In E. Rosch & B.B. Lloyd (Eds.), *Cognition and categorization.* Hillsdale, NJ: Erlbaum.

Child, D. (1970). *The essentials of factor analysis.* London: Holt, Rinehart, & Winston.

Cleeremans, A., & McClelland, J. (1991). Learning the structure of event sequences. *Journa of Experimental Psychology: General, 120,* 235–253.

Dienes, Z., Broadbent, D.E., & Berry, D.C. (1991). Implicit and explicit knowledge bases ir artificial grammar learning. *Journal of Experimental Psychology: Learning, Memory, & Cognition, 17,* 875–887.

Dulany, D.E., Carlson, R.A., & Dewey, G.I. (1984). A case of syntactical learning and judgement: How conscious and how abstract? *Journal of Experimental Psychology: General, 113,* 541–555.

Elio, R., & Anderson, J.R. (1981). The effects of category generalisations and instance similarity on schema abstraction. *Journal of Experimental Psychology: Human Learning ana Memory, 7,* 397–417.

Elman, J. (1990). Finding structure in time. *Cognitive Science, 14,* 179–211.

Ericsson, K.A., & Simon, H.A. (1984). *Protocol analysis: Verbal reports as data.* Cambridge, MA: MIT Press.

Estes, W.K. (1986). Memory storage and retrieval processes in category learning. *Journal oj Experimental Psychology: General, 115,* 155–174.

Estes, W.K., Campbell, J.A., Hatsopoulos, N., & Hurwitz, J.B. (1989). Base-rate effects in category learning: A comparison of parallel network and memory storage-retrieval models. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15,* 556–571.

Gluck, M.A., & Bower, G.H. (1988). Evaluating an adaptive network of human learning. *Journal of Memory and Language, 27,* 166–195.

Hama, D., & Vasburgh, R. (1976). Category breadth and the abstraction of prototypical information. *Journal of Experimental Psychology: Human Learning and Memory, 2,* 322–330.

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

Hinton, G.E. (1987). *Connectionist learning procedures* (Tech. Rep. No. CMV–CS–87–115). Pittsburgh, PA: Carnegie Mellon University, Computer Science Department.

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

Hintzman, D.L. (1990). Human learning and memory: Connections and dissociations. *Annual Review of Psychology, 41,* 109–139.

Howell, D.C. (1987). *Statistical methods for psychology* (2nd ed.). Boston: Duxbury Press.

James, W. (1950). *The principles of psychology.* New York: Dover Publications. (Original work published 1890).

Kemler-Nelson, D.G. (1984). The effect of intention on what concepts are acquired. *Journal of Verbal Learning and Verbal Behavior, 23,* 734–759.

Kemler-Nelson, D.G. (1988). When category learning is holistic: A reply to Ward and Scott. *Memory and Cognition, 16,* 79–84.

Markman, A.B. (1989). LMS rules and the inverse base-rate effect: Comment on Gluck and Bower (1988). *Journal of Experimental Psychology: General, 118,* 417–421.

Massaro, D.W. (1988). Some criticisms of connectionist models of human performance. *Journal of Memory and Language, 27,* 213–234.

Mathews, R.C., Buss, R.R., Stanley, W.B., Blanchard-Fields, F., Cho, J.-R., & Druhan, B. (1989). The role of implicit and explicit processes in learning from examples: A synergistic effect. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 15,* 1083–1100.

McAndrews, M.P., & Moscovitch, M. (1985). Rule-based and exemplar-based classification in artificial grammar learning. *Memory & Cognition, 13,* 469–475.

McClelland, J.L., & Elman, J. (1986). Interactive processes in speech perception: The TRACE model. In J.L. McClelland & D.E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition.* Cambridge, MA: MIT Press.

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.

McClelland, J.L., & Rumelhart, D.E. (1986). A distributed model of human learning and memory. In J. McClelland & D.E. Rumelhart (Eds.), *Parallel distributed processing: Exploring the microstructure of cognition.* Cambridge, MA: MIT Press.

Medin, D.L., Altom, M.W., Edelson, S.M., & Freko, D. (1982). Correlated symptoms and simulated medical classification. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 8,* 37–50.

Medin, D.L., & Edelson, S.M. (1988). Problem structure and the use of base-rate information from experience. *Journal of Experimental Psychology: General, 117,* 68–85.

Medin, D.L., & Schaffer, M.M. (1978). Context theory of classification learning. *Psychological Review, 85,* 207–238.

Medin, D.L., & Schwanenflugal, P.J. (1981). Linear separability in classification learning. *Journal of Experimental Psychology: Human Learning and Memory, 7,* 355–368.

Perruchet, P., & Pacteau, C. (1990). Synthetic grammar learning: Implicit rule abstraction or explicit fragmentary knowledge? *Journal of Experimental Psychology: General, 119,* 264–276.

Posner, M.I., & Keele, S.W. (1968). On the genesis of abstract ideas. *Journal of Experimental Psychology, 77,* 353–363.

Reber, A.S. (1967). Implicit learning of artificial grammars. *Journal of Verbal Learning and Verbal Behavior, 6,* 855–863.

Reber, A.S. (1969). Transfer of syntactic structure in synthetic languages. *Journal of Verbal Learning and Verbal Behavior, 6,* 855–863.

Reber, A.S. (1976). Implicit learning of synthetic languages: The role of instructional set. *Journal of Experimental Psychology: Human Learning and Memory, 2,* 88–94.

Reber, A.S. (1989). Implicit learning and tacit knowledge. *Journal of Experimental Psychology: General, 118,* 219–235.

Reber, A.S., & Allen, R. (1978). Analogic and abstraction strategies in synthetic grammar learning: A functionalist interpretation. *Cognition, 6,* 189–221.

Rescorla, R.A., & Wagner, A.D. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A.H. Black & W.F. Prokasy (Eds.), *Classical conditioning II: Current research and theory.* New York: Appleton-Century-Crofts.

Rumelhart, D.E., & McClelland, J.L. (1986a). On learning the past tenses of English verbs. In J.L. McClelland & D.E. Rumelhart (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition.* Cambridge, MA: MIT Press.

Rumelhart, D.E., & McClelland, J.L. (Eds.). (1986b). *Parallel distributed processing: Explorations in the microstructure of cognition.* Cambridge, MA: MIT Press.

Rumelhart, D.E., & Zipser, D. (1986). Feature discovery by competitive learning. In D.E. Rumelhart & J.L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition.* Cambridge, MA: MIT Press.

Schneider, W. (1987). Connectionism: Is it a paradigm shift for psychology? *Behavioral Research Methods, Instruments, and Computing, 19,* 73–83.

Servan-Schreiber, E., & Anderson, J.R. (1990). Learning artificial grammars with competitive chunking. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 16,* 592–608.

Shanks, D.R. (1990). Connectionism and the learning of probabilistic concepts. *Quarterly Journal of Experimental Psychology, 42,* 209–237.

Stone, G.O. (1986). An analysis of the delta rule and the learning of statistical associations. In D.E. Rumelhart & D.E. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition.* Cambridge, MA: MIT Press.

Venit, S., & Bishop, W. (1985). *Elementary linear algebra* (2nd ed.). Boston: Prindle, Weber, & Schmidt.

Ward, T.B., & Scott, J. (1987). Analytic and holistic modes of learning family resemblance concepts. *Memory and Cognition, 15,* 42–54.

Widrow, C., & Hoff, M.E. (1960). Adaptive switching circuits. *Institute of Radio Engineers, Western Electronic Show and Convention, Convention Record, 4,* 96–104.

Wolff, J.G. (1975). An algorithm for the segmentation of an artificial language analogue. *British Journal of Psychology, 66,* 79–90.

Wolff, J.G. (1977). The discovery of segments in natural language. *British Journal of Psychology, 68,* 97–106.

Wolff, J.G. (1980). Language acquisition and the discovery of phrase structure. *Language and Speech, 23,* 255–269.

APPENDIX A

Relationships Between Models

Appendix A indicates which models made different predictions. The connectionist models are considered first. Correlations relevant to each of the assumptions along which they varied are presented for the Hebb and asymptotic delta rule models. Relationships involving the memory-array models are considered second. For all correlations, $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

Connectionist Models

TABLE 11
Correlations Between Hebb and Asymptotic Delta Rule Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Co-occurrence				
Grammatical	.92	.21	.89	.22
Nongrammatical	.78	.68	.89	.81
Contingency				
Grammatical	.26	.05	.47	.21
Nongrammatical	.48	.48	.54	.54

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

TABLE 12
Correlations Between Single-Letter and Digram Models

	Co-occurrence		Contingency	
	Succ	Sim	Succ	Sim
Hebb				
Grammatical	.97	.98	.98	.98
Nongrammatical	.86	.97	.97	.99
Asymptotic Delta				
Grammatical	.93	.90	.82	.71
Nongrammatical	.87	.84	.77	.85

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.

TABLE 13
Correlations Between Co-occurrence and Contingency Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Hebb				
Grammatical	−.22	.13	−.44	−.01
Nongrammatical	.06	.37	−.15	.26
Asymptotic Delta				
Grammatical	.57	.95	.35	.79
Nongrammatical	.55	.84	.38	.84

Note. $r(\text{crit})_{.05}=.34$, one-tailed, or .40, two-tailed.

TABLE 14
Correlations Between Successive and Simultaneous Models

	Co-occurrence		Contingency	
	Single	Digram	Single	Digram
Hebb				
Grammatical	.96	.96	.91	.92
Nongrammatical	.93	.96	.83	.85
Asymptotic Data				
Grammatical	.45	.60	.56	.16
Nongrammatical	.69	.77	.69	.77

Note. $r(\text{crit})_{.05}=.34$, one-tailed, or .40, two-tailed.

Memory-Array Models

TABLE 15
MINERVA 2 Models

	DI	DC	SI	HC
Grammatical Strings				
DI				.96
DC	.88			.91
SI	.97	.88		.99
SC	.79	.97	.82	.85
Nongrammatical Strings				
DI				.98
DC	.95			.96
SI	.98	.95		.99
SC	.90	.97	.92	.93

Note. I=categorization by intensity; C=categorization by content; S=single-letter coding; D=digram coding. HC=Hebb contingency single-letter simultaneous model as representative of the Hebb contingency models. $r(\text{crit})_{.05}=.34$, one-tailed, or .40, two-tailed.

TABLE 16
Memory-Array Models of Estes (1986)

	ex1	ex2	ex3	ex4	freq
Grammatical Strings					
ex2	.92				
ex3	.68	.41			
ex4	.85	.63	.85		
freq	.41	.12	.86	.72	
HC	.69	.46	.90	.87	.89
Nongrammatical Strings					
ex2	.80				
ex3	.72	.80			
ex4	.83	.84	.97		
freq	.59	.57	.84	.75	
HC	.70	.71	.87	.83	.96

Note. HC=Hebb contingency single-letter simultaneous model, as representative of the Hebb contingency models. $r(\text{crit})_{.05}=.34$, one-tailed, or .40, two-tailed.

APPENDIX B

Asymptotic Delta Rule Models

Appendix B indicates how it was determined that the asymptotic delta rule models actually achieved asymptotic weights. If a model was trained on linearly independent strings, then it should be able to complete each of them perfectly, that is, **p** should be identical to **a**. A standard procedure given in elementary linear algebra textbooks (e.g., Venit & Bishop, 1985) was used to determine the linear dependence of the training strings. A matrix was formed in which the coded strings formed columns. Thus, for single-letter models the matrix was 30×20, and for digram models it was 72×20. The row-reduced echelon form of the matrix was determined by Jordan-Gauss elimination (using a Basic program suggested by Venit & Bishop, 1985). If the row-reduced echelon form of the matrix contains only elementary columns, then the strings are linearly independent, otherwise they are linearly dependent and the nature of the dependence is indicated by the row-reduced matrix (see Venit & Bishop, 1985).

For the digram models (regardless of whether they were of the co-occurrence or contingency type), the acquisition strings were linearly independent. Thus, **p** should be identical to **a** for all acquisition strings for the asymptotic simultaneous models, if they are truly asymptotic. Indeed, for the contingency model, this was true to six decimal places; and for the co-occurrence model, this was true to three decimal places. Thus, the simultaneous digram models can be regarded as actually asymptotic. What about

the successive digram models? In this case, the linear independence of the acquisition strings is not relevant, because each feature is predicted only by features in previous positions. Thus, these models will be able to reproduce the acquisition strings perfectly only if the feature at each position is a linearly separable function of features in the previous position. Because this is certainly not true of features in the first position (the first letter can be one of two letters), these models will never be able to complete the training strings perfectly. In fact, for the contingency successive model, the average correlation between **p** and **a** over all acquisition strings was .88 (range .83–.95). However, because this correlation had remained constant to at least two decimal places over at least the last 100 iterations of learning, the successive digram models were also regarded as actually asymptotic.

For the single-letter models (regardless of whether they were of the co-occurrence or contingency type), the acquisition strings were linearly dependent. Specifically, considering the ordering of the acquisition strings given in Table 1, String 14 (i.e., VXRRRR, call it e14) can be obtained by subtracting String 7 (MVRXRR, e7) from String 6 (MVRX, e6), and adding to String 12 (VXRR, e12); similarly, $e17 = e2 - e1 + e8 - e10 + e15$, and $e18 = e4 - e1 + e8 - e10 + e15$. Thus, the single-letter models will never be able to complete the acquisition strings perfectly. However, the correlations between **p** and **a** remained constant to at least two decimal places over at least the last 100 iterations of learning, and so the single-letter models were also regarded as actually asymptotic.

For the pre-asymptotic models, the correlation between **p** and **a** were less than for their asymptotic counterparts. For example, the correlations for the pre-asymptotic contingency digram simultaneous model ranged between .62 and .87, whereas for the asymptotic model they were virtually unity. Nonetheless, the rank orderings of string difficulty were comparble between asymptotic and pre-asymptotic models; Table 17 displays the correlations in rank order of string difficulty between asymptotic and pre-asymptotic models.

TABLE 17
Correlations Between Asymptotic and Pre-asymptotic Delta Rule Models

	Single		Digram	
	Succ	Sim	Succ	Sim
Co-occurrence				
Grammatical	.91	.75	.90	.76
Nongrammatical	.80	.87	.93	.89
Contingency				
Grammatical	.80	.86	.90	.63
Nongrammatical	.79	.88	.94	.91

Note. $r(\text{crit})_{.05} = .34$, one-tailed, or .40, two-tailed.