CHAPTER 11

An Analysis of the Delta Rule and the Learning of Statistical Associations

G. O. STONE

The development of parallel distributed processing models involves two complementary enterprises: first, the development of complete models with desired operating characteristics; and second, the in-depth analysis of component mechanisms and basic principles. The primary objective in modeling is the development and testing of complete systems. In general these models are complex and their behavior cannot be fully deduced directly from their mathematical description. In such cases, simulation plays an important role in understanding the properties of a model. Although simulations are useful in determining the properties of a specific model, they do not, on their own, indicate how a model should be modified when a desired behavior is not achieved. An understanding of basic principles and a collection of potential mechanisms with known properties provide the best guides to the development of complex models.

This chapter provides an analysis of one of the most popular components—namely, the error correction learning rule developed by Widrow and Hoff (1960). This learning rule which has been analyzed and employed by a number of authors (Amari, 1977a, 1977b; Kohonen, 1974, 1977; Sutton & Barto, 1981), has been called the Widrow-Hoff rule by Sutton and Barto (1981) and is generally referred to as the delta rule in this book. This rule is introduced in Chapter 2, discussed extensively and generalized in Chapter 8, and employed in models discussed in a number of chapters—most notably Chapters 17 and 18. In the present chapter I show how concepts from linear algebra

and vector spaces can provide insight into the operation of this learning mechanism. I then show how this mechanism can be used for learning statistical relationships between patterns, and finally show how the delta rule relates to multiple linear regression. Concepts from linear algebra are used extensively; for explanation of these concepts, especially as applied to PDP models, the reader is referred to Chapter 9.

The Delta Rule in Vector Notation

The delta rule is typically applied to the case in which pairs of patterns, consisting of an input pattern and a target output pattern, are to be associated so that when an input pattern is presented to an input layer of units, the appropriate output pattern will appear on the output layer of units. It is possible to represent the patterns as *vectors* in which each element of the vector corresponds to the activation value of a corresponding unit. Similarly, we can represent the connections from input units to the output units by the cells of a weight matrix. For linear units, the output vector can be computed by multiplying the input vector by the weight matrix. In the present chapter our analysis is restricted to linear units. (See Chapter 8 for a discussion of the delta rule for nonlinear units.)

Now we imagine a learning situation in which the set of input/output pairs are presented (possibly repeatedly) to the system. If the set of input vectors are orthogonal (i.e., at right angles to each other), a simple pattern associator can be constructed using a *product* learning rule in which the change in weight w_{ji} following the presentation of pattern p is given by the product of the ith input element and the jth target element, that is,

$$\Delta_p w_{ji} = t_{pj} i_{pi}$$

where t_{pj} represents the value of the desired or target output for the jth element of pattern p and i_{pi} is the activation value of the ith element of the input for that pattern.[1] In vector notation, we can write the change for the entire weight matrix as

$$\Delta_p = \mathbf{t}_p \mathbf{i}_p^T$$

[1] Note this is essentially the *Hebbian* learning rule. In the Hebbian rule it is assumed that the product of the activation levels of the input and output units determine the weight change. If we assume that the activation of the output unit is entirely determined by the target input the product rule described here is identically the Hebbian rule.

where, as usual, bold letters indicate vectors, uppercase indicates matrices and the superscript T indicates the transpose of a vector or matrix. This learning rule was described in some detail in Chapter 9 and that discussion will not be repeated here. It was shown there that if the input vectors are normalized in length so that $i_p \cdot i_p = 1$ and are orthogonal, the product rule will, after the presentation of all of the input/output patterns, lead to the following weight matrix:

$$\mathbf{W} = \sum_{p=1}^{P} \mathbf{t}_p \mathbf{i}_p^T.$$

If the input vectors are orthogonal, there will be no interference from storing one vector on others already stored so that the presentation of input $\mathbf{i}_p$ will lead to the desired output $\mathbf{t}_p$, that is,

$$\mathbf{W}\mathbf{i}_p = \mathbf{t}_p$$

for all patterns p from 1 to P. Unfortunately, we cannot always insure that the input vectors are orthogonal. Generally, the storage of one input/output pair can interfere with the storage of another and cause crosstalk. For this case a more sophisticated learning rule is required.

Fortunately, as we saw in Chapter 8, the delta rule is a rule that will work when the input patterns are *not* orthogonal. This rule will produce perfect associations so long as the input patterns are merely *linearly independent* (see Chapter 9) and will find a weight matrix which will produce a "least squares" solution for the weight matrix when an exact solution is not possible (i.e., the input patterns are not linearly independent). In matrix notation the rule can be written as

$$\mathbf{W}(n) = \mathbf{W}(n-1) + \eta \delta(n) \mathbf{i}^T(n) \tag{1}$$

where $\mathbf{W}(n)$ is the state of the connection matrix after n presentations, $\mathbf{i}(n)$ is the input presented on the nth presentation, η is a scalar constant which determines the rate of learning, and $\delta(n)$ is the difference between the desired and actual output on trial n, such that

$$\delta(n) = \mathbf{t}(n) - \mathbf{W}(n-1)\mathbf{i}(n) \tag{2}$$

where $\mathbf{t}(n)$ is the desired output (or *target*) for presentation n and $\mathbf{W}(n-1)\mathbf{i}(n) = \mathbf{o}(n)$ is the output actually produced on that presentation. $\mathbf{W}(0)$ is assumed to be the matrix with all zero entries. In other words, the weight matrix is updated by adding the outer product of the response error and the input. (See Chapter 9 for discussion of outer product.) Proofs concerning the convergence of this recursion to the optimum weight matrix (in the sense outlined above) are provided by Kohonen (1974, 1977, 1984).

The Delta Rule in Pattern-Based Coordinates

To this point we have discussed the delta rule for what Smolensky (Chapter 22) has called the *neural* or *unit* level of representation. Before proceeding, it is useful to consider the form that the rule takes in the *conceptual* level of representation in which there is one vector component for each concept. In general, the input and output patterns correspond to an arbitrary set of vectors. Interestingly, it is possible to show that the delta rule applies only to the "structure" of the input and output vectors and not to other details of the representation. In a linear system, it is only the pattern of correlations among the patterns that matter, not the contents of the specific patterns themselves.

We can demonstrate this by deriving the same learning rule following a *change of basis* from the *unit basis* to the *pattern basis*. Since a detailed discussion of the process whereby bases can be changed is given in Chapter 9 and, in more detail, in Chapter 22, I will merely sketch the concept here. Each pattern over a set of units corresponds to a vector. If there are N units, then the vector is of dimension N. In the unit basis, each element of the vector corresponds to the activation value of one of the units. Geometrically, we can think of each unit as specifying a value on a dimension and the entire vector as corresponding to the coordinates of a point in N-dimensional space. Thus, the dimensions of the space correspond directly to the units (this is why it is called the unit basis). Now, a change of basis amounts essentially to a change in coordinate system. This can be accomplished through rotation, as well as other linear transformations. Converting to the pattern basis merely involves transforming the coordinate system so that the patterns line up with the axes. Figure 1 illustrates a simple case of this process. In Figure 1A we give the geometric representation of the patterns. Pattern 1, $\mathbf{p}_1$, involves two units, each with activation value $+1$. Pattern 2, $\mathbf{p}_2$, has activation values $<+1,-1>$. The patterns described in the unit basis are

$$\mathbf{p}_1 = \begin{bmatrix} +1 \\ +1 \end{bmatrix} \quad \text{and} \quad \mathbf{p}_2 = \begin{bmatrix} +1 \\ -1 \end{bmatrix}.$$

Figure 1B shows the same two vectors, but now expressed with respect to a new coordinate system, the *pattern* coordinate system. In this case the axes correspond to the *patterns* not the *units*. The vectors corresponding to patterns 1 and 2 now become

$$\mathbf{p}^*_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \text{and} \quad \mathbf{p}^*_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

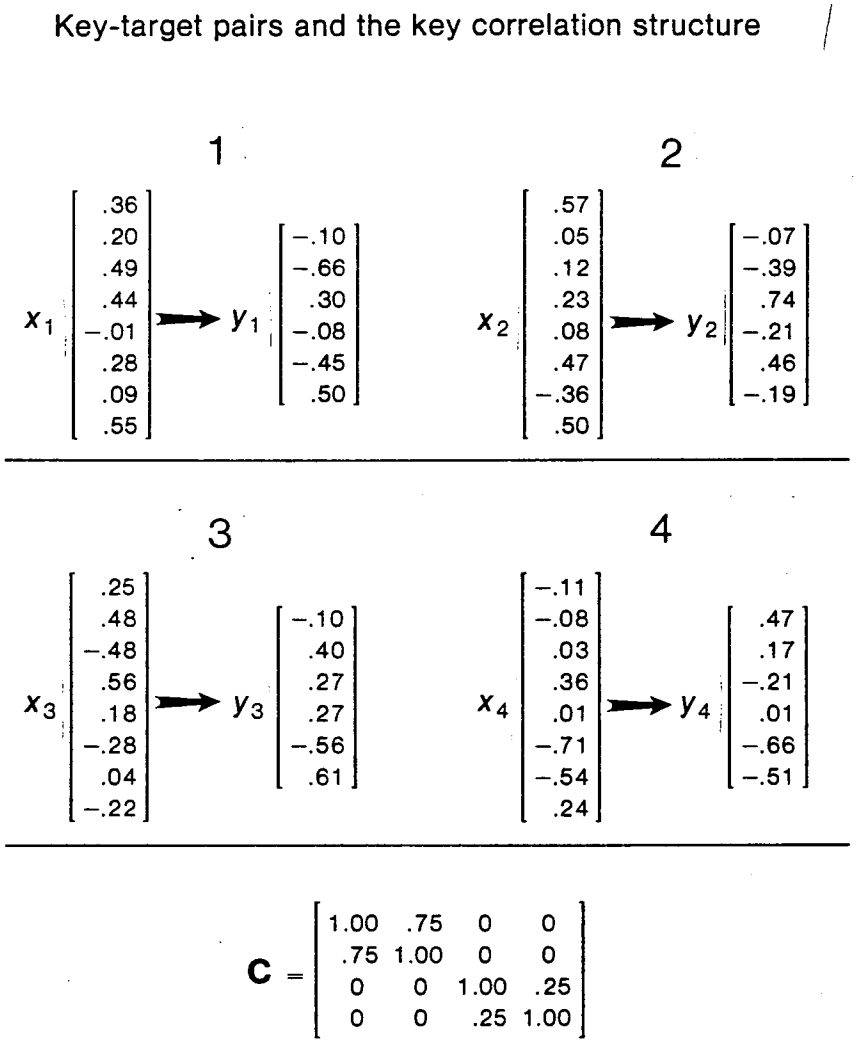

FIGURE 1. An example of conversion from unit-based coordinates into pattern-based coordinates.

In general, conversion to a new basis requires a matrix **P** which specifies the relationship between the new and old coordinate systems. For each vector, $\mathbf{p}_i$, we write the new vector $\mathbf{p}^*_i$ as $\mathbf{p}^*_i = \mathbf{P}\mathbf{p}_i$. If all of the vectors and matrices of the original system are converted into the new basis, we simply have a new way to describe the same system. For present purposes we have two transformation matrices, one that

transforms the input patterns into a coordinate space based on the input patterns, which we denote $\mathbf{P}_I$, and one that transforms the target patterns into a coordinate space based on the target patterns, $\mathbf{P}_T$. In this case, we have $\mathbf{i}^*_i = \mathbf{P}_I \mathbf{i}_i$ for the input vectors and $\mathbf{t}^*_i = \mathbf{P}_T \mathbf{t}_i$ for the target vectors. Moreover, since the output vectors must be in the same space as the target vectors we have $\mathbf{o}^*_i = \mathbf{P}_T \mathbf{o}_i$. We must also transform the weight matrix $\mathbf{W}$ to the new basis. Since the weight matrix maps the input space onto the output space, both transformations must be involved in transforming the weight matrix. We can see what this transformation must be by considering the job that the weight matrix must do. Suppose that in the old bases $\mathbf{Wi} = \mathbf{o}$ for some input $\mathbf{i}$ and output $\mathbf{o}$. In the new bases we should be able to write $\mathbf{W}^*\mathbf{i}^* = \mathbf{o}^*$. Thus, $\mathbf{W}^*\mathbf{P}_I\mathbf{i} = \mathbf{P}_T\mathbf{o}$ and $\mathbf{P}_T^{-1}\mathbf{W}^*\mathbf{P}_I\mathbf{i} = \mathbf{o} = \mathbf{Wi}$. From this we can readily see that $\mathbf{P}_T^{-1}\mathbf{W}^*\mathbf{P}_I = \mathbf{W}$ and finally, we can write the appropriate transformation matrix for $\mathbf{W}$ as

$$\mathbf{W}^* = \mathbf{P}_T \mathbf{W} \mathbf{P}_I^{-1}.$$

We can multiply both sides of Equation 1 by $\mathbf{P}_T$ on the right and $\mathbf{P}_I^{-1}$ on the left. This leads to

$$\mathbf{P}_T\mathbf{W}\mathbf{P}_I^{-1}(n) = \mathbf{P}_T\mathbf{W}\mathbf{P}_I^{-1}(n-1) + \mathbf{P}_T \eta \delta(n) \mathbf{i}^T(n) \mathbf{P}_I^{-1}$$

which, by substitution, can be written as

$$\mathbf{W}^*(n) = \mathbf{W}^*(n-1) + \eta\delta^*(n)\left[\mathbf{P}_I^{-1}\mathbf{i}^*(n)\right]^T \mathbf{P}_I^{-1},$$

where

$$\delta^*(n) = \mathbf{t}^*(n) - \mathbf{W}^*(n-1)\mathbf{i}^*(n). \tag{3}$$

Finally, by rearranging we have

$$\mathbf{W}^*(n) = \mathbf{W}^*(n-1) + \eta\,\delta^*(n)\mathbf{i}^*(n)^T\mathbf{C} \tag{4}$$

where the matrix $\mathbf{C}$, given by $\mathbf{C} = (\mathbf{P}_I^{-1})^T\mathbf{P}_I^{-1}$, is a matrix which holds the correlational information among the original input patterns. To see this, recall that we are changing the input patterns into their pattern basis and the target patterns into their pattern basis. Therefore, the vector $\mathbf{i}^*_j$ consists of a 1 in the jth cell and zeros everywhere else. Thus, since $\mathbf{i}_j = \mathbf{P}_I^{-1}\mathbf{i}^*_j$, we see that $\mathbf{P}_I^{-1}$ must be a matrix whose jth column is the jth original input vector. Therefore, $\mathbf{C}$ is a matrix with the inner product of the input vectors $\mathbf{i}_i$ and $\mathbf{i}_j$ occupying the ith row and jth column. This inner product is the vector correlation between the two patterns.

We have finally constructed a new description which, as we shall see, allows many insights into the operation of the delta rule which are normally obscured by the internal structure of the patterns themselves. Instead, we have isolated the critical interpattern structure in the matrix **C**.

One advantage of this new description is that the output the system actually produces—even when it does not match any target exactly—can easily be interpreted as the weighted average of the various target patterns. The value in each cell of the output vector is the coefficient determining the amount of that target in the output. In this case the sum squared error for input/output pattern p, given by

$$E_p = \sum_j (t^*_j - o^*_j)^2,$$

measures the error directly in terms of the degree to which each *target pattern* is present in the output, rather than the degree to which each *unit* is present. It should be noted, of course, that this new pattern-based error function is related to the old unit-based error by the same change of basis matrices discussed above.

It might be observed further that under this description, the perfect associator—which results when the input and output patterns are linearly independent—will be the identity matrix, **I**, in which the main diagonal has a 1 in each entry and all other entries are 0. It should be noted, however, that the preceding analysis of this new description has assumed the input and target output patterns were linearly independent. If they are not, no such pattern basis exists. However, there is an analogous, but somewhat more complex, development for the case in which these vectors cannot form a legitimate basis.

I will now demonstrate some of the useful insights which can be gained through this analysis by comparing the unit and pattern basis descriptions for a sample learning problem. Figure 2A gives the representations of the four input/output patterns to be learned in the unit basis. Figure 2B gives the representations in the pattern basis along with the correlation matrix **C** required in the new description. These patterns are all linearly independent and were generated under the constraint that each pattern has unit length and that the input patterns have the correlation structure given in Figure 2B. Clearly the description given in the pattern basis is simpler.

Figure 3 shows the states of **W** and **W*** after one, four, and eight sweeps through the four input/output patterns. While inspection of the unit-based representations gives no direct information about the degree of learning and crosstalk between targets, this information is explicit in the pattern-based representation. For example, one can discern that the error for the pairs with highly correlated inputs (pairs 1 and 2) is

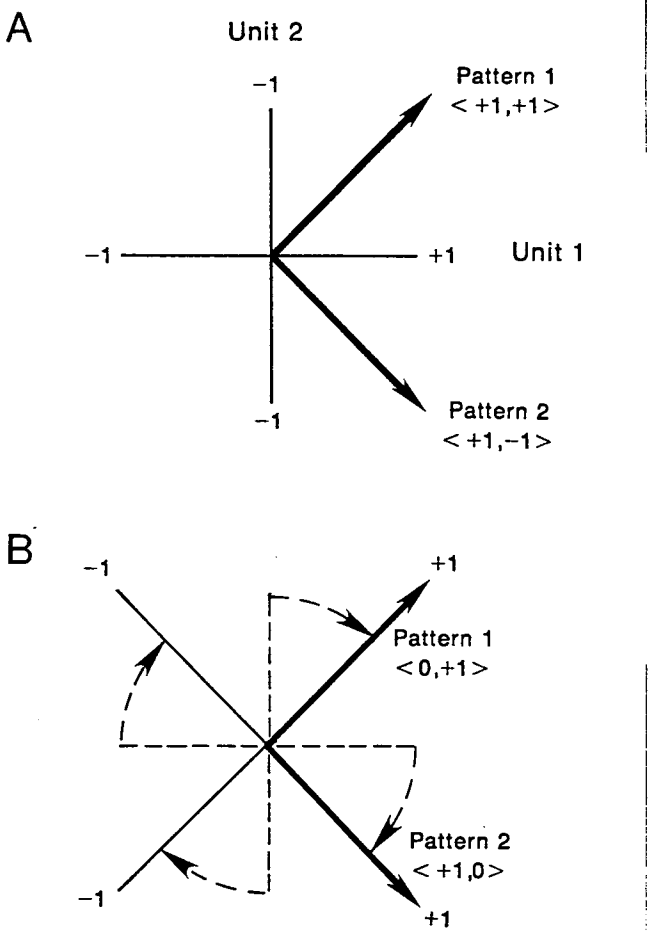

FIGURE 2. Comparison of the unit-based and pattern-based descriptions of a sample learning problem.

greater at each stage than that for the pairs with slightly correlated input patterns (pairs 3 and 4). Moreover, there is no intrusion of targets associated with orthogonal inputs. In addition, the intrusion of targets from correlated pairs is least for the pair most recently learned, pairs 2 and 4. (The patterns were presented in order 1-2-3-4 on each sweep.) Finally, it is clear from inspection of the pattern-based weight matrix that after eight sweeps the patterns have been almost perfectly learned.

The pattern-based formulation also allows a more detailed analysis of the general effect of a learning trial on the error. We can define the "potential error" to pattern j, $\delta_j{}^*$ as

Learning with g = 1.20

AFTER 1 Learning Cycle

Pattern Based: msec=0.09				Unit Based: msec=0.049							
0.39	-0.18	0.00	0.00	-0.12	-0.13	0.02	0.10	-0.01	-0.42	-0.35	0.12
0.90	1.20	0.00	0.00	-0.03	0.08	-0.59	0.00	0.12	-0.29	-0.18	-0.40
0.00	0.00	1.11	-0.06	0.57	0.28	0.08	0.35	0.10	0.52	0.03	0.33
0.00	0.00	0.30	1.20	-0.04	0.13	-0.23	0.07	0.05	-0.14	0.10	-0.23
				0.30	-0.33	0.17	-0.59	-0.04	0.94	-0.13	0.23
				0.05	0.50	-0.17	0.20	0.06	0.20	0.81	-0.42

AFTER 4 Learning Cycles

Pattern Based: msec=0.00				Unit Based: msec=0.00							
0.98	-0.01	0.00	0.00	-0.15	-0.18	0.07	0.01	-0.04	-0.34	-0.31	0.13
0.03	1.01	0.00	0.00	-0.07	0.02	-0.57	-0.08	0.10	-0.28	-0.20	-0.41
0.00	0.00	1.00	0.00	0.59	0.13	-0.30	0.11	0.16	0.50	-0.29	0.13
0.00	0.00	0.00	1.00	-0.05	0.16	-0.10	0.14	0.02	-0.15	0.17	-0.16
				0.36	-0.42	-0.47	-0.81	0.08	0.86	-0.53	-0.12
				0.06	0.66	0.16	0.53	0.02	0.09	1.01	-0.19

AFTER 8 Learning Cycles

Pattern Based: msec=0.00				Unit Based: msec=0.00							
1.00	0.00	0.00	0.00	-0.15	-0.18	0.07	0.01	-0.04	-0.34	-0.31	0.13
0.00	1.00	0.00	0.00	-0.07	0.02	-0.57	-0.08	0.10	-0.28	-0.20	-0.41
0.00	0.00	1.00	0.00	0.59	0.13	-0.32	0.10	0.16	0.50	-0.30	0.12
0.00	0.00	0.00	1.00	-0.05	0.16	-0.10	0.14	0.02	-0.15	0.18	-0.15
				0.35	-0.43	-0.49	-0.82	0.08	0.85	-0.54	-0.13
				0.06	0.66	0.17	0.54	0.02	0.10	1.01	-0.18

FIGURE 3. Comparison of unit-based and pattern-based weight matrices after one, four, and eight learning cycles.

$$\delta_j{}^*(n) = t^*_j - \mathbf{W}^*(n) i^*_j. \tag{5}$$

Substituting for $\mathbf{W}^*(n)$ from Equation 1, gives

$$\delta_j{}^*(n) = t^*_j - \mathbf{W}^*(n-1) i_j - \eta \delta_k{}^* i^*_k{}^T \mathbf{C} i^*_j$$

where k is the index of the pattern presented on trial $n-1$. Simplifying further, we have the recursive form:

$$\delta_j{}^*(n) = \delta_j{}^*(n-1) - \eta \delta_k{}^*(n-1) \mathbf{i}^*_k{}^T \mathbf{C} \mathbf{i}^*_j. \tag{6}$$

Since the vectors $\mathbf{i}^*_j$ and $\mathbf{i}^{*T}_k$ consist of a 1 and the rest zeros, the entire expression $\mathbf{i}^{*T}_k \mathbf{C} \mathbf{i}^*_j$ reduces to c_{kj}, the entry in the kth row and jth column of matrix $\mathbf{C}$. Thus, Equation 6 becomes simply

$$\delta_j^*(n) - \delta_j^*(n-1) = -\eta c_{kj} \delta_k^*(n-1). \tag{7}$$

In other words, the decrease in error to the jth input/output pair due to a new learning trial is a constant times the error pattern on the new learning trial. The constant is given by the learning rate, η, times the correlation of the currently tested input and input from the learning trial. Thus, the degree to which learning affects performance on each test input is proportional to its correlation with the pattern just used in learning. Note that if η is small enough, the error to the presented pattern always decreases. In this case Equation 7 can be rewritten

$$\delta_k^*(n) = \delta_k^*(n-1)(1 - \eta c_{kk}).$$

Recalling that c_{kk} is given by $\mathbf{i}^T_k \mathbf{i}_k$, the length of the kth input vector, we can see that the error will always decrease provided $|1 - \eta \mathbf{i}^T_k \mathbf{i}_k| < 1$.

To summarize, this exercise has demonstrated that a mechanism can often be made more conceptually tractable by a judicious transformation. In this case, expressing the possible input and output representations in the appropriate pattern bases clarified the importance, indeed the sufficiency, of the input "structure" (i.e., the pattern of inner products among the input vectors) in determining the role of the input representations in learning. Furthermore, converting the weight matrix into a form from which the errors at any stage of learning can be read directly allowed us to "see" the learning more obviously. The result has been a clearer understanding of the operation of the delta rule for learning.

STATISTICAL LEARNING

In this section we extend our analysis of the delta rule from the case in which there is a fixed target output pattern for each input pattern to the case in which sets of input patterns are associated with sets of output patterns. We can think of the sets as representing *categories* of input and outputs. Thus, rather than associate particular input patterns with particular output patterns, we analyze the case in which *categories* of input patterns are associated with *categories* of output patterns. This, for example, might be the case if the system is learning that dogs bark. The representation for dog might differ on each learning trial with respect to size, shagginess, etc. while the representation for the bark

might vary with regard to pitch, timbre, etc. In this case, the system is simultaneously learning the categories of *dog* and *bark* at the same time it is learning the association between the two concepts.

In addition, when we have category associations, statistical relationships between the input and output patterns within a category can be picked up. For example, the system could learn that small dogs tend to have high-pitched barks whereas large dogs may tend to have low-pitched barks.

In order to analyze the case of statistical learning, we now treat the input/output pairs of patterns as random variables. In other words, each time pattern i_j is selected as input, its entries can take different values. Similarly, the target output for pair j, t_j will have variable entries. The probability distributions of these random variables may take any form whatsoever, but they are assumed not to change over time. Moreover, we can consider the entire set of input/output pairs to form a single probability distribution. We then assume that on each trial an input/output pair is randomly sampled from this overall probability distribution.

We proceed with our analysis of statistical learning by computing the *expected* or *average* change in the weight matrix following a presentation. From Equations 1 and 2 we get the following form of the delta rule:

$$\mathbf{W}(n) = \mathbf{W}(n-1) + \eta[\mathbf{t}(n) - \mathbf{W}(n-1)\mathbf{i}(n)]\mathbf{i}^T(n).$$

Simplifying and taking the expected value of each side we have

$$E[\mathbf{W}(n)] = E[\mathbf{W}(n-1)](\mathbf{I} - \eta E[\mathbf{i}(n)\mathbf{i}^T(n)]) + \eta E[\mathbf{t}(n)\mathbf{i}^T(n)]. \quad (8)$$

Note, we may take

$$E[\mathbf{W}(n-1)\mathbf{i}(n)\mathbf{i}^T(n)] = E[\mathbf{W}(n-1)]E[\mathbf{i}(n)\mathbf{i}^T(n)]$$

since each trial is assumed to be statistically independent of all preceding trials, upon which $\mathbf{W}(n-1)$ depends. Letting $\mathbf{R}_I = E[\mathbf{i}\mathbf{i}^T]$ be the pattern of statistical correlations among the input patterns and $\mathbf{R}_{IO} = E[\mathbf{t}\mathbf{i}^T]$ be the statistical correlations between the input and target patterns, we can rewrite Equation 7 as

$$E[\mathbf{W}(n)] = E[\mathbf{W}(n-1)](\mathbf{I} - \eta\mathbf{R}_I) + \eta\mathbf{R}_{IO}.$$

If we solve the recursion by replacing $\mathbf{W}(n-1)$ with an expression in terms of $\mathbf{W}(n-2)$ etc. down to $\mathbf{W}(0)$ and assuming that $\mathbf{W}(0) = 0$, the matrix of all 0 entries, we can write the expected value of the weight matrix after n trials as

$$E[\mathbf{W}(n)] = \eta \mathbf{R}_{IO} \sum_{j=0}^{j=n} (\mathbf{I} - \eta \mathbf{R}_I)^j. \tag{9}$$

Fortunately, in the limit, this matrix reduces to a simpler form. To see this, we must introduce the concept of the *pseudo-inverse* of a matrix. This is a matrix which, unlike the inverse, is certain to exist for all matrices, but which has a number of properties in common with an true inverse. (See Chapter 9 for a discussion of matrix inverses and the conditions under which they exist.) In particular, it *is* the true inverse, if the true inverse exists. The *pseudo-inverse* of a matrix $\mathbf{B}$, designated $\mathbf{B}^+$, is given by

$$\mathbf{B}^+ = \eta \mathbf{B}^T \sum_{j=1}^{\infty} (\mathbf{I} - \eta \mathbf{B}\mathbf{B}^T)^j$$

provided η is sufficiently small. (See Rao & Mitra, 1971, and Kohonen, 1977, 1984, for a full discussion of the pseudo-inverse.)

In order to convert Equation 9 into a form that includes the expression for the pseudo-inverse, we observe that since the square matrix $\mathbf{R}_I = E[\mathbf{i}\mathbf{i}^T]$ has independent rows and columns, we can select a matrix $\mathbf{P}$ such that $\mathbf{P}\mathbf{P}^T = \mathbf{R}_I$ and $\mathbf{P}$ also has linearly independent rows and columns. Since the generalized inverse of $\mathbf{P}$, $\mathbf{P}^+$, is also the true inverse of $\mathbf{P}$, it satisfies $(\mathbf{P}^T)^{-1}\mathbf{P}^T = \mathbf{I}$. Thus, taking the limit as $n \to \infty$ of Equation 9 and substituting $\mathbf{P}$, we can write

$$\lim_{n\to\infty} E[\mathbf{W}(n)] = E[\mathbf{W}_\infty] = \mathbf{R}_{IO}(\mathbf{P}^T)^{-1}[\eta \mathbf{P}^T \sum_{j=1}^{\infty} (\mathbf{I} - \eta \mathbf{P}\mathbf{P}^T)^j]. \tag{10}$$

Now, by substituting in for the pseudo-inverse of $\mathbf{P}$ and simplifying we get

$$E[\mathbf{W}_\infty] = \mathbf{R}_{IO}(\mathbf{P}^T)^{-1}\mathbf{P}^+ = \mathbf{R}_{IO}(\mathbf{P}\mathbf{P}^T)^{-1} = \mathbf{R}_{IO}\mathbf{R}_I^{-1}. \tag{11}$$

Since the rows and columns of $\mathbf{R}_I$ are linearly independent, $\mathbf{R}_I^{-1} = \mathbf{R}_I^\dagger$. So we finally get

$$E[\mathbf{W}_\infty] = \mathbf{R}_{IO}\mathbf{R}_I^\dagger. \tag{12}$$

Now we wish to show that, after training, the system will respond appropriately. Without further restrictions, we can demonstrate a minimal appropriateness of the response, namely, we can show that $E[\mathbf{W}_\infty \mathbf{i}] = E[\mathbf{t}]$. In other words, we can show that the mean output of the system, after learning, is the mean target. Since the test trials and learning trials are statistically independent we can write

$$E[\mathbf{W}_\infty \mathbf{i}] = E[\mathbf{W}_\infty]E[\mathbf{i}].$$

Now, substituting in from Equation 9 we have

$$E[\mathbf{W}_\infty \mathbf{i}] = \mathbf{R}_{IO}\mathbf{R}_I^{\dagger} E[\mathbf{i}] = E[\mathbf{t}\mathbf{i}^T(\mathbf{i}\mathbf{i}^T)^+]E[\mathbf{i}].$$

Although it is not generally true that $(\mathbf{BC})^+ = \mathbf{C}^+\mathbf{B}^+$, this relation does hold for $\mathbf{B} = \mathbf{i}$ and $\mathbf{C} = \mathbf{i}^T$, where $\mathbf{i}$ is a column vector. Thus, we have

$$E[\mathbf{W}_\infty \mathbf{i}] = E[\mathbf{t}(\mathbf{i}^+\mathbf{i})^T(\mathbf{i}^+\mathbf{i})].$$

Finally, since $\mathbf{i}$ has only one column, its columns are linearly independent and $\mathbf{i}^+\mathbf{i} = 1$. We have therefore obtained the desired result.

Thus far we have only shown that the mean response to inputs is equivalent to the mean of the target patterns. This result says nothing about the appropriateness of the response to a particular pattern. Ideally we would want the expected response to a particular pattern to yield the expected value of our target given the input. We can show that the input will produce this result as long as $\mathbf{i}$ and $\mathbf{t}$ are distributed normally with zero means. Although this seems to be a strong assumption, it is not a difficult situation to obtain. First, we can easily arrange that the input patterns have zero means by simply having a *bias* feeding into each unit equal to minus the mean of the value for that cell of the pattern. This is not especially difficult, but we will not dwell on the process here. (See Chapter 8 for a discussion of *biases* and the learning of biases.) Suffice it to say that it is not very difficult to convert a set of input vectors into a set of patterns with zero mean.

The requirement of normal distributions is often not as restrictive as it appears. When input patterns being associated are themselves the output of a linear system, each entry in the pattern will be a linear combination of the original input's entries. If the patterns have large dimensionality (i.e., there are many components to the vectors), one obtains an approximation to an infinite series of random variables. A powerful central-limit theorem due to Lyapunov (Eisen, 1969, Ch. 13) shows that such a series will converge to a normal distribution so long as several weak assumptions hold (most importantly, the means and variances of each random variable must exist and none of the random variables may be excessively dominant).

Under these conditions, it can be shown that the expected value of the target $\mathbf{t}$ given the input $\mathbf{i}$, takes the form $E[\mathbf{t}|\mathbf{i}] = \mathbf{R}_{IO}\mathbf{R}_I^{-1}\mathbf{i}$ (Meditch, 1969, chap. 3). Since $E[\mathbf{W}_\infty] = \mathbf{R}_{IO}\mathbf{R}_I^{-1}$, we have shown that

$$E[\mathbf{W}_\infty \mathbf{i}] = E[\mathbf{t}|\mathbf{i}], \tag{13}$$

so that after a sufficient number of learning trials, the law of large numbers and the convergence of the delta rule learning process imply

that, given a particular input, the system will produce an output equal to the average of the targets paired with that input. In this sense, systematic covariation of input/output pairs will be learned.

The Delta Rule and Multiple Linear Regression

Some readers may have already noticed the similarity of the learning task we have been analyzing to the problem encountered in multiple linear regression. In a linear regression problem the objective is to predict, to the degree possible, one variable, say y, from a set of variables $\mathbf{x}$. In these problems we typically wish to find a set of coefficients, $\mathbf{b}$, such that

$$\hat{y}_j = b_0x_{0j}+b_1x_{1j}+b_2x_{2j} \cdots b_nx_{nj} = \mathbf{b}^T\mathbf{x}_j$$

(where x_0 is taken to be 1) and the sum-squared error

$$\sum_{j=1}^{n} (y_j - \hat{y}_j)^2$$

is minimized. This is precisely the problem that the delta rule seeks to solve. In this case, each element of the target vector for input/output pair (p $\mathbf{t}_p$) is analogous to a to-be-predicted observation y_j; our prediction variables $\mathbf{x}_j$ are analogous to our input vectors i_p; our regression coefficients $\mathbf{b}$ correspond to a row of the weight matrix $\mathbf{W}$; and the intercept of the regression line, b_0, corresponds to the *bias* often assumed for our units (cf. Chapter 8). In our typical case the target vectors have many components, so we are simultaneously solving a multiple regression problem for each of the components of the target vectors. Now, the standard result from linear regression, for zero-mean random variables, is that our estimate for the vector $\mathbf{b}$, $\hat{\mathbf{b}}$ is given by

$$\hat{\mathbf{b}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

where $\mathbf{X}$ is the matrix whose columns represent the values of the predictors and whose rows represent the individual observations. (Again, we take the first column to be all 1s.) Now, note from Equation 12 that the delta rule converges to

$$E[\mathbf{W}_\infty] = (E[\mathbf{i}^T\mathbf{i}])^+E[\mathbf{i}^T\mathbf{t}].$$

This equation is the strict analog of that from linear regression theory.[2] If we assume that each output unit has a bias corresponding to the intercept b_0 of the regression line, we can see that the delta rule is, in effect, an iterative method of computing the best, in the sense of least squares, linear regression coefficients for our problems.

SUMMARY

To summarize, this chapter has shown that close examination of the delta rule reveals a number of interesting and useful properties. When fixed patterns are being learned, the rule's operation can be elucidated by converting from a unit-based description to a pattern-based description. In particular, the analysis showed that the correlations between the input patterns, and not the specific patterns used, determined their effect on the learning process. Thus, any alteration of the specific input patterns that does not alter the correlations will have no effect on learning by a linear delta rule. It was also shown that expressing the inputs and outputs in terms of the patterns being learned facilitated analysis of the learning process by allowing one to read directly from the output produced the degree to which each target was present in the output generated by a given input pattern.

When the patterns being learned are variable, it was noted that the final weight matrix could be expressed simply in terms of the intercorrelations among the input patterns, $\mathbf{R}_I$, and the correlations between the input and output patterns, $\mathbf{R}_{IO}$. It was also shown that when several reasonable requirements for the distribution of the input/output random variables are met, the delta rule will learn the pattern of covariation between the inputs and targets. Finally, we showed that the delta rule carries out the equivalent of a multiple linear regression from the input patterns to the targets. Those familiar with linear regression should conclude from this both the power of the rule and its weaknesses. In particular, wherever a linear regression is insufficient to provide a good account of the relationship between input and target patterns, the system will perform poorly. The solution to this problem is to have nonlinear units and intermediate layers of hidden units. Chapter 8 is a detailed discussion of the generalized delta rule and its application to these situations.

---

[2] Actually, there is a slight difference in convention between our development and that typical of linear regression. In our case, the stimulus vectors are the column vectors, whereas in linear regression the predictor variables are the rows of the matrix $\mathbf{X}$. Thus this equation differs by a transposition from Equation 12. This has no consequences for the points made here.

The preceding discussion does not, by any means, provide a complete analysis of the delta rule. Rather, it illustrates two important ideas. First, that a basic principle (in this case, the use of pattern-based, rather than unit-based representations) can provide valuable insights into the operation of a useful mechanism; and second, that the analysis of component mechanisms which were designed for one use can often reveal new applications.