CHAPTER 7

Modeling Cognitive Processes: The Interactive Activation Model

In this chapter our goal is to consider the application of PDP modeling techniques to the task of accounting for human cognitive processes, as revealed through psychological experimentation. As our example for this, we've chosen the interactive activation model of word perception (McClelland & Rumelhart, 1981; Rumelhart & McClelland, 1982). This model exemplifies our approach to modeling psychological processes, and it is of tractable size for running on smaller machines.

BACKGROUND

Our initial interest in parallel distributed processing mechanisms grew out of an attempt to capture our ideas about continuous, interactive processes, particularly as they applied to the problems of visual word recognition and reading. Both of us had already done both experimental and theoretical work in this area, but without the benefit of simulations (see McClelland 1976, 1979; Rumelhart, 1977; Rumelhart & Siple, 1974).

Our primary aim was to account for contextual influences on perception. These influences have been described since psychologists first began to present visual or auditory stimuli under controlled conditions (Bagley, 1900; Cattell, 1886). Among the early observations was the fact that subjects could identify far more letters from a single brief flash if the letters fit together to form a word than if the letters made a random string. Context could override the sensory input too, as in the cases where subjects reported the strong impression that they saw all the letters in the word *FOREVER* when in fact *FOYEVER* was shown (Pillsbury, 1897).

In early studies the experimenter relied on what is generally called a free report of the contents of the briefly displayed stimulus, and many researchers pointed out that there were serious methodological problems with this. It has often been pointed out that subjects might see as much in the two cases, but forget less or correctly guess more when the stimuli form words.

An experiment that controlled for both guessing and forgetting at once was carried out by Reicher (1969) and followed up by a number of investigators, including Wheeler (1970), Johnston and McClelland (1973, 1974, 1980; Johnston, 1978; McClelland & Johnston, 1977), and several others (Baron & Thurston, 1973; Manelis, 1974; Massaro, 1973; Massaro & Klitzke, 1979; Spoehr & Smith, 1975).

In Reicher's experiments, a target was presented (e.g., *E*) either in a word (e.g., *READ*), in a scrambled letter string (e.g., *AEDR*), or in isolation. The presentation was followed by a masking stimulus, which consisted of a jumbled array of letter parts, and a pair of letters, which was keyed to the position occupied by the target letter. One of the letters was the target letter itself, and the other was another letter that fit the context (if any) to make an item of the same type. For the displays *READ, AEDR*, and *E* in isolation, the pair could be *E* and *O*, presented with a row of dashes to indicate which display location was being tested:

```
  E
- - - -
  O
```

The subject's task was to choose which of the two letters had appeared in the indicated position. The target could appear in any of the four positions, and subjects did not know in advance which position would be tested on a given trial.

Reicher's test is called the *forced choice test*. Using this test, he found that subjects were more accurate when the letters occurred in words than in either of the other two conditions. This finding is called the *word superiority effect*.

Reicher's finding is important because it indicates that the advantage for words is not simply a matter of guessing letters that fit the context better from fragmentary cues. Rather, it appears that the perceptual system is better able to use the information in the display when the letters form a word with their context. The fact that the advantage holds for words over single letters makes it difficult to view the phenomenon as a result simply of forgetting, since a single letter surely places a very light load on memory.

Reicher's findings, backed up by a large literature of further experimental tests, seemed to us to be a very clear demonstration that context plays a role in perception. We therefore set out to model this phenomenon, basing our approach on a number of basic assumptions.

Basic Assumptions of the Interactive Activation Model

Here we describe each of the basic tenets of the interactive activation model and explain why we adopted each one.

Perception occurs in a multilevel processing system. This assumption is nearly ubiquitous, and so we will give it little discussion; surely there are separate levels of representation for visual features, for words, and for larger wholes such as sentences. For our model of the processing of individual words or strings of letters, we assumed that there are at least three levels: a visual feature level, a letter level, and a word level.

Deeper levels of processing are accessed via intermediate levels. This assumption has often seemed contentious, particularly with respect to visual word recognition. We have assumed that a letter level is interposed between the feature and the word level because words appear to be defined, not in terms of their particular visual configurations, but in terms of the sequences of letters that they contain. Thus *READ*, *read*, and *rEaD* are all recognizable as words, and letters in such stimuli are all perceived better than letters in unrelated context (e.g., the *E* in *rEaD* is perceived better than the *E* in *aEdR*; cf. Adams, 1979; McClelland, 1976). Thus it would appear that readers can use their knowledge of words to perceive sequences of letters, even if the visual configuration of the input is highly novel.

If, as we assume, access to the word level is via the letter level, then sequences of letters should be more effective as masks for words than sequences of feature bundles that do not form letters. This prediction was confirmed by Johnston and McClelland (1980).

Processing is interactive. By this we mean that processing involves the simultaneous consideration of both bottom-up input information and top-down knowledge-based constraints. Our principle reason for this belief was the well-known and ubiquitous role of contextual factors in perceptual processing already alluded to above. We take the role of word context in letter perception as one example of this kind of interactive processing. Models that captured the *outcome* of the simultaneous consideration of bottom-up and top-down information had been developed by others (particularly Morton, 1969), but we wished to embody this assumption in a dynamic processing model.

Information flow is continuous. At the time we began to consider interactive processing, the predominant view among psychologists working in perceptual information processing was that information processing occurred through a sequence of discrete steps. Each step took a certain amount of time and resulted in a discrete output. However, alternatives to this view were developed during the course of the 1970s (cf. McClelland, 1979; Norman & Bobrow, 1976; Turvey, 1973). In fact, the utility of continuous information flow was pointed out quite early in the *pandemonium* model of Selfridge (1955), an early AI model designed to account for the role of context in letter recognition. For us, the assumption of continuity seemed to be required in order to capture contextual influences in word recognition (McClelland, 1976; Rumelhart, 1977). The reason is that if the word level is to influence processing at the letter level, then the letter level must be making information available to the word level before processing at the letter level is complete.

PDP models as a way of capturing these basic assumptions. We turned to PDP models because they provided a simple and direct way of making our basic assumptions about continuous, interactive processing explicit in a computational model. By assuming a processing unit for each possible hypothesis about the input at each of the three levels of processing, by allowing each unit to be working continually, updating its own activation and sending activation to other units, and by allowing units to influence each other via simple excitatory and inhibitory interactions, we found we were able to capture our basic assumptions in a simulation model and explore how well these assumptions could account for contextual influences in letter perception.

Central Questions

In developing the interactive activation model, there were several basic questions:

- Could we make a PDP embodiment of our basic assumptions account for the basic fact that word context facilitates letter perception, as established by Reicher and others?

- Could we account for the fact that subjects perceive letters in pronounceable nonwords (e.g., *REAT*) more accurately than letters in random or scrambled strings and, under some conditions, more accurately than single letters (Johnston & McClelland, 1973; Wheeler, 1970)?

- Could we apply the model to the large body of existing data and show that we could really account for the existing findings? The most important facts we considered were these: (a) The perceptual advantage for letters in words is shared with pronounceable nonwords; that is, letters in words and in pronounceable nonwords show a sizeable advantage over single letters or letters in random strings. (b) Within pronounceable words and nonwords, there is no consistent advantage for strings containing frequent letter clusters (e.g., *PEEP* or *TEEP*) compared to those containing much less frequent letter clusters (e.g., *POET* or *HOET*). Though apparent letter-cluster effects are found in some studies (see Rumelhart & McClelland, 1982, Experiment 9), other studies did not show these effects (McClelland & Johnston, 1977). Our hope was to account for both patterns of results, based on detailed aspects of the particular materials used in different experiments. (c) For letters in words, under the visual conditions in which Reicher's word superiority effect was obtained, there is no advantage for letters occurring in contexts that strongly constrain the identity of the letter (e.g., the *C* in *CLUE*: only three letters make words in the context *_LUE*) compared to letters in context that exert much weaker constraints (e.g., the *C* in *CAKE*; 10 letters make words in the context *_AKE*; Johnston, 1978). Again, however, such effects do occur in other studies. Our hope was to use the model to understand and account for these differences.

- Could we account for a set of new findings from our own laboratory? These findings were based on the use of a new technique for visual presentation, in which the letters in a four-letter string could be started and ended at different times. We found (as reported in Rumelhart & McClelland, 1982) that subjects perceived a particular letter better when the other letters with which it occurred were presented for a longer time. This was true both when the letter formed a word with the context and when it formed a pronounceable nonword with the context, but not when the letter was embedded in a random-letter string.

The approach that we took in trying to answer these questions was to begin by trying to develop a model that produced the basic perceptual facilitation advantage for letters in words when compared to letters in nonword strings; this turned out to be one of the hardest parts of the project. We tried a number of variants on the basic activation equation, as well as a wide range of different parameter values before we developed enough understanding for what we were doing to find a combination of assumptions that worked. Once we accomplished this, we began to consider the list of phenomena we wanted to account for, working our way more or less down the list just given. Our goal was to find a single formulation, together with a single set of parameter values, that would allow us to give a fairly close account of the findings discussed earlier.

Two further aspects of our approach are worth mentioning. First, we endeavored to keep the model as simple as possible, within the constraint that we preserved sufficient structure to capture our basic assumptions. For example, we used a highly simplified representation of the visual forms of letters, and we assumed that each letter in a visual display came rigidly channeled into one of four letter positions. Second, we did not attempt to obtain detailed quantitative fits of the model to the results of particular experiments. Rather, we attempted to come as close as we could to producing results that captured the major qualitative features of the phenomena.

We do not mean to suggest that detailed quantitative fits are not appropriate in many cases. Rather, we want to suggest that in this and many other cases, detailed quantitative fits may require very detailed and specific assumptions (for example, about the confusability of particular pairs of letters) that fall outside of the basic assumptions that are at issue. Attention to such assumptions may in certain instances interfere with the search for understanding of the basic principles that transcend such details. Under these circumstances, a simplified model may yield a more satisfying explanatory account, even when it is known to be wrong in some details, if it provides an explanation for the qualitative patterns observed in the empirical phenomena. (More discussion of this point may be found in Sejnowski's discussion of the role of computational models in *PDP:21*, pp. 387-389.)

We were also concerned with making sure we understood exactly what was going on in the model that allowed it to account for the phenomena. To this end, we spent a great deal of time studying the processing of individual examples so that the details of what was happening in the simulations would be clear. This kind of detailed study of individual items will be the focus of this chapter. In assessing the model, we supplemented this individual example approach by running large simulations with lists of items taken from the original experiments we were trying to understand. We omit these kinds of simulations here, although they played a central role in evaluating how well the model could account for the facts reported in particular experiments.

When we felt we had been reasonably successful in accounting for the phenomena that we wanted to account for, we began to consider whether there might not be additional experiments that we might do to test the principles underlying the approach. We were able to come up with one such experiment (Rumelhart & McClelland, 1982, Experiment 10). It will be described in more detail in one of the exercises.

THE INTERACTIVE ACTIVATION MODEL

In this section we describe the essential features of the interactive activation (IA) model. Some additional details may be found in McClelland and Rumelhart (1981).

Network Architecture

The model consists of units at each of three processing levels: the feature level, the letter level, and the word level (see Figure 1). At the feature level, there is a set of units that serves to detect features in each of four letter positions. Within each set, there is a unit for the *presence* of each of the line segments in the simple font used by Rumelhart and Siple (1974) (shown in Figure 2) and another unit for the *absence* of each such line segment. These units are said to be detectors for the different values (present, absent) of each of the possible line-segment features. This assumption allows the model to distinguish between not knowing whether a line segment is present (both units off) and knowing that a segment is not present (the absence unit on and the presence unit off). At the letter level, there are four sets of letter units, one for each position. Each set contains a unit for each of the 26 letters of the English alphabet. At the word level, there is a single set of detectors for each word in a list of 1179 four-letter words taken from the word list of Kucera and Francis (1967). The set

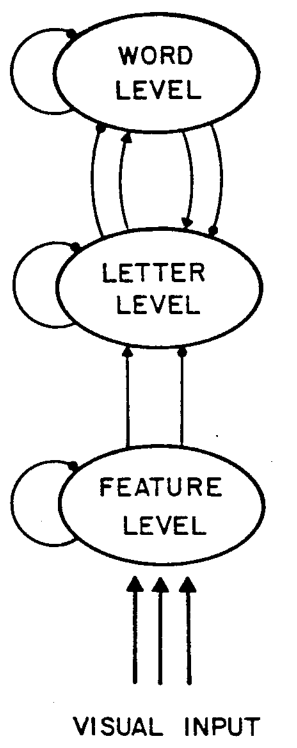

FIGURE 1. The basic architecture of the interactive activation model. (From "An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings" by J. L. McClelland and D. E. Rumelhart, 1981, *Psychological Review*, *88*, 375-407. Copyright 1981 by the American Psychological Association. Reprinted by permission.)

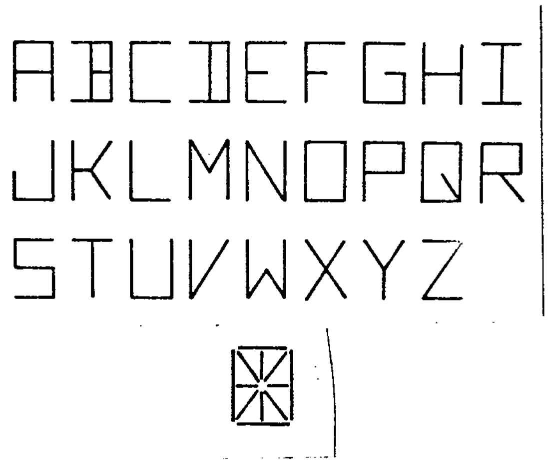

FIGURE 2. The Rumelhart-Siple letter font used by the interactive activation model. (From "Process of Recognizing Tachistoscopically Presented Words" by D. E. Rumelhart and P. Siple, 1974, *Psychological Review*, *81*, 99-118. Copyright 1974 by the American Psychological Association. Reprinted by permission.)

includes all the words with frequencies of two or more per million, excluding proper names, contractions, abbreviations, and foreign words that crept into the count.

Each unit can be taken as representing the hypothesis that the entity it stands for—feature, letter, or word—is present in the input. The activations of the units are monotonically related to the strengths of these hypotheses, according to a function that will be described later.

Connections

The connections among the units are intended to encode the mutual constraints among hypotheses about the possible contents of a four-letter display. The overall framework allows excitatory connections between units on different levels that are mutually consistent and allows inhibitory connections between units on different levels that are mutually inconsistent. For example, *T* in the first letter position is mutually consistent with all words beginning with *T*; these units therefore have mutually excitatory connections. To simplify matters, some of these connections are left out of the model: First, there are no feedback connections that are inhibitory. Second, there is no feedback at all from the letter level to the feature level. This leaves the following sets of between-level connections:

- *Feature-to-letter excitation.* Feature units have excitatory connections to all the letter units in the same spatial position that contain the feature. Thus, the presence unit for a horizontal bar at the top of a letter position is connected to all the letters in this same position that have this feature.

- *Feature-to-letter inhibition.* Feature units have inhibitory connections to all of the letter units in the same spatial position that do not contain the feature. Thus each feature unit, when active, excites some of the letters and inhibits all the others.

- *Letter-to-word excitation.* Each letter unit has excitatory connections to each of the units standing for a word that contains the letter in the appropriate position. Thus the unit for *T* in the first position excites *TAKE* but not *CART.*

- *Letter-to-word inhibition.* Each letter unit has inhibitory connections to each of the units standing for a word that does not contain the letter in the appropriate position. Thus the unit for *T* in the first position inhibits all words that do not begin with *T*, including *CART*, *STOP*, and so on.

- *Word-to-letter excitation.* Each word unit has excitatory connections to the units for all of the letters in the word. Thus, there is an excitatory connection from *TAKE* to *T* in the first position, to *A* in the second, and so on.

In addition to these between-level connections, there are also within-level connections. These are exclusively inhibitory and are used to implement competition among mutually exclusive interpretations of the same portion of the input. All of the units for letters in the same spatial position are mutually inhibitory; and at the word level, all of the word units are mutually inhibitory.

The Concept of a Trial

In the IA model, we simulate trials from psychological experiments. A trial consists of a sequence of fields presented one after the other for processing. Before the trial, network activations are reset to baseline levels, and the cycle number is set to 0. The first field is presented at the beginning of cycle 1, followed by the second field, if any, at some later time, and so on. The last field is assumed to stay on until after a response occurs. Note that it is possible to present a field of blanks, if this is desired.

During processing, interactive activation processes govern the activations of units in the network. In addition, a readout process is assumed to be operating concurrently with the processing activity itself. The results of this readout process are assumed to be available for overt report.

After the processing of the sequence of fields has gone on for some chosen number of cycles, there may be a two-alternative forced-choice test of the kind used by Reicher and others; that is, the model may be confronted with a position cue and two letters, with the task of indicating which had actually occurred in the corresponding letter position.

Input to the Model

The inputs to the model—the contents of the fields presented during the processing trials—consist of specifications of which feature-level units should be turned on. This turning on may be deterministic (i.e., all features specified may be turned on) or it may be stochastic (specified features may be "detected" by the model with some probability). This probability can vary from field to field to allow for the possibility that the display is more or less legible in different cases.

Processing in the Model

Processing in the model occurs via the interactive activation and competition mechanisms discussed in Chapter 2. For simplicity, however, the feature units are treated as binary. Each feature unit is turned on or off by external input, and none of the feature units receive any input from any other units.

Letter and word units are activated according to the IAC equations given in Chapter 2. We show them again here for convenience. The net input to each unit is given by

$$net_i = \sum_j w_{ij} o_j . \qquad (1)$$

Recall that o_j, the output of unit j, is equal to the extent to which the activation of the unit exceeds its threshold $(a_j - t_j)$, or 0 if $a_j \leqslant t_j$. The net input acts as a force that drives the activation of the unit upward if the net input is excitatory and downward if it is inhibitory:

If $(net_i > 0)$,

$$\Delta a_i = (max - a_i) netinput_i - decay (a_i - rest).$$

Otherwise,

$$\Delta a_i = (a_i - min) netinput_i - decay (a_i - rest).$$

Readout From the Model

Readout from the model is thought of as a separate process that integrates the activations of the units over time to assess the *response strength* of each unit. The response strength is defined to be the model's measure of the strength of the hypothesis that the entity a unit stands for is present in the input. The readout process chooses one of the units probabilistically, based on its response strength relative to the strength of other units.

The response strength of each unit is given by

$$s_i(t) = e^{k\bar{a}_i(t)} \tag{2}$$

where k is a scale factor and where $\bar{a}_i$, the running average of the activation of unit i, is given by

$$\bar{a}_i(t) = (orate)a_i(t) + (1 - orate)\bar{a}_i(t-1). \tag{3}$$

(Initially or when the model is reset, $\bar{a}_i(0) = a_i(0) = rest_i$, the resting activation of unit i.) Following Luce's (1959) choice model, the probability of choosing a particular item i as the response at time t is simply

$$p(r_{i,t}) = \frac{s_i(t)}{\sum_{j \in C} s_j(t)}. \tag{4}$$

This probability is called the *response probability* of response i at time t. Here, C is the set of competing alternatives (all letters in the same position for letter responses; all words for word responses), including unit i itself. Note that this response choice rule has the effect of ensuring that the response probabilities always sum to 1 for each set of competing alternatives.

The Forced-Choice Test

In simulating what happens in the forced-choice test, we first made the assumption that choices were based on responses read out from the letter level only. Under this assumption, the word superiority effect is due to the feedback from the word level to the letter level.

Because the choice alternatives appear after the target display has been replaced by the mask, we assumed that readout from the network had to occur without regard to the alternatives. That is, we assumed that the subjects did their best to identify all the letters in the target display following the response choice assumptions described above, and only after doing so did they consult the forced-choice alternatives.

The determination of response probabilities in the model is complicated by the fact that the response strengths on which they are based rise and fall with time. Some assumptions must be made about the timing of readout. We assumed that subjects chose a time after onset of the target display that optimized their probability of choosing correctly. In practice, this means that readout occurs just before the onset of the postdisplay mask, if there is one. When no mask is used, readout is assumed to occur after activation and response strengths reach their asymptotic values.

Once readout has occurred, the rule for making choices is very simple: If the response letter chosen for the target position matches one of the alternatives, that alternative is chosen; otherwise, the choice is a random guess. From this rule, the probability of correctly choosing the target letter is the probability that this letter was chosen by the response readout process, plus 0.5 times the probability that neither the correct nor the incorrect alternative was chosen by the readout process.

Although the readout process is assumed to be stochastic, we do not actually simulate the probabilistic choice between alternatives. We calculate the probability of reading out the correct and incorrect alternatives and use these probabilities to calculate the probability correct in the forced choice.

Parameters

Here we discuss the parameters of the interactive activation model. There are parameters that influence the internal processing dynamics, parameters that reflect assumptions about the input, and parameters that influence the assignment of response probabilities during the readout process. The list of parameters of the model is rather long, but the majority are fixed at rather arbitrary values. The main parameters modified to capture the basic experimental effects we wished to account for are the excitatory and inhibitory strength parameters, *alpha* and *gamma*. However, in keeping with our philosophy of allowing users the greatest degree of control over the programs as possible, we have made all of the following parameters modifiable, as we shall see later.

alpha and *gamma*

The excitatory and inhibitory connection strength parameters, *alpha* and *gamma*, respectively, depend only on the processing levels of the units in question. This means, for example, that the strengths of the excitatory connections from feature units to letter units are the same for all such excitatory connections. The model has a separate parameter for feature-to-letter excitation, another for letter-to-word excitation, and another for word-to-letter excitation, as well as parameters for feature-to-letter inhibition, letter-to-letter inhibition, letter-to-word inhibition, and word-to-word inhibition. In our simulations, these parameters were subject to tuning, with the goal of obtaining the best possible fit to the entire ensemble of experimental data we were considering. The values we settled on in this process are shown in Table 1.

decay

The model provides separate parameters for the rate of decay, at both the letter and the word levels. In practice, however, the decay parameters were both set to the same value (0.07) at a fairly early point in our simulations, and then were left at this value for the remainder of our experiments with the model. This value was chosen because it seemed to be the largest value that would nevertheless allow the model to settle smoothly to a target pattern of activation. With larger values the activations of units can start to oscillate wildly from cycle to cycle.

threshold

The model provides separate parameters for the output thresholds of units at the letter and word levels. In fact, at the word level there are separate threshold values for output to the letter level and for inhibitory output to other words. These thresholds were generally left at 0, except during our simulations of the contextual enhancement effect in pronounceable nonwords (see Ex. 7.4).

max and *min*

The model also provides separate parameters for the maximum and minimum activations units are allowed to have at the letter and word levels. We have always left the maximum activations at 1.0.

TABLE 1

DEFAULT VALUES FOR THE ALPHA AND GAMMA PARAMETERS USED IN THE IA MODEL

Excitation parameters (*alpha*):	
feature to letter	0.005
letter to word	0.07
word to letter	0.30
Inhibition parameters (*gamma*):	
feature to letter	0.15
letter to word	0.04
word to word	0.21
letter to letter	0.00

We did, however, experiment with different values for the minimum, settling on −0.20 for both the letter and the word levels.

rest and *fgain*

The model provides separate parameters for the resting activation levels of units at the letter and word levels. After some experimentation, the resting levels were generally left at 0. However, in the case of words, it should be noted that the value of the word-level resting activation parameter was not 0 for all units, but was offset downward from 0 depending on the word's frequency. The most frequent word known to the model was *that*, which was assigned a resting activation offset of 0.00. Other words were given resting activation offsets ranging between −0.92 and −0.01, according to a function that assigned offsets proportional to the log of the frequency of the word, subtracted from the log of the frequency of *that*. The model multiplies these offsets by the value of a frequency-scale parameter, *fgain*, and subtracts the result from 0. Throughout the simulations we used a value of 0.05 for *fgain*. Thus, the resting levels of words actually ranged from 0.00 to nearly −0.05.

oscale

This parameter corresponds to the parameter k in Equation 2. The model provides separate parameters for scaling the output strengths of units at the letter and word levels. A larger value of *oscale* is needed at the word level to compensate for the fact that there are more competitors at this level of processing, even though most of them usually are assigned highly negative activation values by the model. We recommend that the user keep the given values of 10.0 for letter-level output and 20.0 for word-level output.

fdprob

For each display field the user wishes to present, it is possible to set a separate value for the probability that features are detected from the input. By default, *fdprob* is set to 1.0, which means that all of the features of the input are detected. However, this parameter can be set to a lower value to simulate the effects of degraded visual presentation.

estr

Many experiments find that end letters are perceived more accurately than letters internal to a word. To accommodate this, we provide separate parameters for the strength of feature-level activations for each letter position. By default, these parameters are set to 1.0, and we recommend that users leave them at these values unless they specifically wish to explore these positional differences.

orate

This last parameter determines the rate of accumulation of activation for the purpose of determining response strength. Its default value is 0.05. Generally, we have not found the value to be terribly critical, although we have generally assumed that it is small so that activations are translated into outputs only gradually.

Processing Under Different Visual Display Conditions

In our simulations, we tended to lump visual display conditions used in different experiments into two categories, based on the different conditions used by Johnston and McClelland (1973). One condition was called the *bright-target/pattern-mask* condition and the other was called the *dim-target/blank-mask* condition.

For the bright-target/pattern-mask condition we assumed that the target display was bright and clear enough so that all visual features of the display were detected and that the same applied to the patterned masking stimulus that followed the target. We further assumed that the effect of the mask was to quickly clear out the pattern of activation at the letter level, replacing it with a new pattern.

These assumptions led us to assume that the feature-to-letter inhibition was very strong, so that features of the mask would quickly inhibit letter activations that had been produced by the target display. A side effect of this was that no letters received net bottom-up excitatory input unless they were consistent with all of the features in a particular display position. As a result, under bright-target/pattern-mask conditions, only the letter actually displayed ever became activated on the basis of bottom-up information. Therefore, we found that we had little need for letter-to-letter inhibition. Consequently, though the model provides for the possibility of such inhibitory influences, we set the letter-to-letter inhibition parameter to 0.

Given that all the features of the display are detected, why is it that performance is less than perfect in the forced-choice test? The answer is simply that it takes time for activation to build up and be read out. The role of feedback is to enhance the activation of letters and, therefore, to increase the probability of correct read out from the letter level.

For the dim-target/blank-mask conditions, we assumed that the temporal brightness summation between the target and the blank mask operated so that the display was approximately equivalent to a very low-contrast, and hence degraded, input. In this situation, we assumed that visual feature information could only be detected imperfectly. The trial is simulated as a single display of letters with a feature detection probability considerably less than 1. The effect of this is that several letters generally are consistent with the detected features in each letter position. Under these conditions, the role of feedback from the word level is to selectively enhance the activations of units at the letter level that fit together with active letters in other positions to form words or to activate groups of words.

IMPLEMENTATION

Data Structures

The implementation of the interactive activation model in the **ia** program is similar to the implementation of the IAC model, although it differs from it in many details. One important difference is that the connections among the units are not in fact specified in a connection matrix. Instead, they are determined by table look-up. There are two relevant tables: the *word* table and the *uc* table.

The *word* table, as its name implies, contains a list of all of the words known to the model, stored as a sequence of four lowercase ASCII characters. To determine whether a particular letter unit should activate a particular word, the model checks to see if the letter is in the word in the appropriate position. We do not, of course, assume that activation in the mind is actually done by table look-up.

The *uc* table contains a list of all the features of the letters. The table is called *uc* to remind the user that the model only knows one alphabet and that is the uppercase Rumelhart-Siple alphabet shown in Figure 2. Each row of the *uc* table consists of fourteen 1s and 0s, indicating whether the corresponding character does or does not have the corresponding segment from the Rumelhart-Siple font in it. For example, the row corresponding to the letter *A* is

1,1,1,1,1,0,1,0,1,0,0,0,0

The exact arrangement of the features will be described when we explain the use of the program. The table also contains several special, nonletter characters, in addition to the uppercase letters. These will be described later.

The *uc* table is used both to specify the set of input features, given a display specification consisting of a sequence of letters entered by the user, and to determine which letter units should be activated when a particular feature unit is activated.

One other important data structure is the *trial* data structure. This is simply a list of field onset times and their contents. This information is specified via the *trial* command, which will be described in the section on using the program.

Processing

As in the **iac** program, processing is controlled by the *cycle* routine. Here is what it looks like: