

Chapter 5

The relevance of learning procedure

Learning plays a central role in connectionist research. The knowledge needed to perform a task must be encoded in terms of weights on connections between units in a network. For tasks that involve fairly simple constraints between inputs and outputs, it is sometimes possible to analytically derive a set of weights that is guaranteed to cause the network to settle into good solutions (Hopfield, 1982; Hopfield & Tank, 1985). However, for tasks involving more complex relationships between inputs and outputs, such as mapping orthography to phonology via semantics, correct behavior requires such highly-complex interactions among weights that it becomes infeasible to hand-specify them. In this case, it is necessary to rely on a learning procedure that takes these interactions into account in deriving an appropriate set of weights.

Many alternative procedures can be used to train a network on a particular task. Most of them take the form of optimization. The task is re-expressed in terms of an *objective* or *error* function that defines a numeric measure of how well the network is performing the task overall. This measure is usually defined in terms of the unit states that are produced in processing each input. For instance, a common error function is the squared difference between the states produced by the network and the correct states, summed over each output unit and each training example. If the actual states match the correct states exactly, the error is zero and the task is solved. If the actual states are quite different from what they should be, the error will be high. Since the states of units are actually caused by the weights, the error function can be thought of as defining a measure of how well the current set of weights solves the task. The learning procedure determines how to change the weights so as to reduce the error, and thus solve the task better.

Although the error on a task is the result of the interaction of all the weights, the crux of most learning procedures is a simplification that calculates how each weight in the network should be changed to reduce the error *assuming the rest of the weights remain fixed*. A natural way to change the weight is in proportion to its influence on the error—that is, in proportion to the partial derivative of the error with respect to the weight. Although the weight changes are calculated as if other weights will not change, if they are small enough their collective effect is guaranteed to (very slightly) reduce the overall error.

In understanding this procedure, it helps to think of a high-dimensional space with a dimension for each weight. This may be easiest to imagine for a network with only two weights. Each point in this space—a plane in two dimensions—defines a set of weights that produces some amount of error if used by the network. If we represent this error along an additional dimension corresponding to height, then the error values of all possible weight sets forms an *error surface* in weight space. A good set of weights has low error and corresponds to the bottom of a “valley” in this surface. At any stage in learning, the network can be thought of as being at the point on the error surface “above” the point for the current set of weights, with a height given by the error for those weights. Possible weight changes consist of movements in different directions along the surface. Changing each weight in proportion to its error derivative amounts to moving in the direction of steepest descent.

Often learning can be accelerated by using the error derivatives in more complex ways in determining how far and in what direction to move in weight space. A common technique is to add a proportion of the previous weight change into the current one—this has the effect of introducing *momentum* to movement in weight space (Plaut et al., 1986). A similar effect is obtained by giving each weight a separate, adaptive learning rate (Jacobs, 1988). “Second-order” methods (le Cun, 1988; Fahlman, 1988) estimate the curvature of the error surface based on successive updates in order to safely take larger steps in weight space. More powerful techniques, such as “conjugate gradient,” maintain much more information about previous weight changes. These approaches can be quite effective at speeding up learning in some circumstances but could not be plausibly implemented using the local information available to individual units in a network. However, to a large extent the issues regarding the application of these acceleration methods can be separated from those concerning the calculation of the error derivatives themselves.

The most widespread procedure for computing error derivatives in connectionist networks is back-propagation (Bryson & Ho, 1969; le Cun, 1985; Parker, 1985; Rumelhart et al., 1986a; 1986b; Werbos, 1974). This procedure is quite powerful because it explicitly calculates the derivatives by propagating information from the output units backward to earlier parts of the network (and earlier iterations in time) according to the chain rule. The power and generality of back-propagation has dramatically extended the applicability of connectionist networks to problems in a wide variety of domains. However, this power also raises concerns about its appropriateness for the purposes of modeling in cognitive psychology and neuropsychology. In particular, the procedure uses information in ways that seem neurophysiologically implausible—a straightforward implementation of the procedure would require error signals to travel backward through synapses and axons (Crick, 1989; Grossberg, 1987). As such, it seems unlikely that back-propagation is what underlies human learning, and thus its use in modeling the *results* of human learning is somewhat suspect.

Proponents of the use of back-propagation in cognitive modeling have replied to this argument

in two ways. The first is to demonstrate how the procedure might be implemented in a neurophysiologically plausible way (e.g. Parker, 1985). The more common approach, and the one adopted by H&S, is to argue that back-propagation is only one of a number of procedures for performing gradient descent learning in connectionist networks. As such it is viewed merely as a “programming technique” for developing networks that perform a task, and is not intended to reflect any aspect of human learning *per se*. The implicit claim is that back-propagation develops representations that exhibit the same properties as would those developed by a more plausible procedure, but does it much more efficiently. However, this claim is rarely substantiated by a demonstration of the similarity between systems developed with alternative procedures.¹

In this chapter we attempt to replicate the main results obtained thus far with back-propagation, within the more plausible learning framework of contrastive Hebbian learning in a deterministic Boltzmann Machine (DBM). Following a brief description of the framework, we define an architecture for mapping orthography to phonology via semantics similar to those used with back-propagation. After training the network, we compare its behavior under a variety of lesions and with that of the back-propagation networks. We also develop a closely-related stochastic GRAIN network and compare it with the deterministic one. In addition to being more plausible as procedures that might underly human learning, both DBM and GRAIN networks have interesting computational characteristics not shared by the back-propagation networks. We conclude the chapter by demonstrating how these characteristics are useful for understanding two aspects of deep dyslexic reading behavior: greater confidence in visual vs. semantic errors, and preserved lexical decision.

5.1 Deterministic Boltzmann Machines

Deterministic Boltzmann Machines (Peterson & Anderson, 1987; Hinton, 1989b) were originally derived as approximations to stochastic Boltzmann Machines (Hinton & Sejnowski, 1983). However, in order to simplify the presentation we will describe only the deterministic version. The units in a DBM are closely related to those in a back-propagation network (see Appendix 10). The state $s_i^{(t)}$ of each unit i at time t is a non-linear function of its summed input.

$$s_i^{(t)} = \lambda s_i^{(t-1)} + (1 - \lambda) \tanh \left(\frac{1}{T} \sum_j s_j^{(t-1)} w_{ij} \right) \quad (5.1)$$

Unit states change somewhat “sluggishly,” so that the new state is a weighted average (with proportion $\lambda = 0.6$ for our simulations) of the old state and the contribution from the new input. The hyperbolic tangent function “tanh” is the symmetric version of the sigmoid function, ranging

¹Terry Sejnowski (personal communication) has successfully re-implemented NETalk (Sejnowski & Rosenberg, 1987), a feed-forward back-propagation network that maps orthography to phonology, as a stochastic Boltzmann Machine. However, he made no direct comparisons of the representations that the two procedures developed.

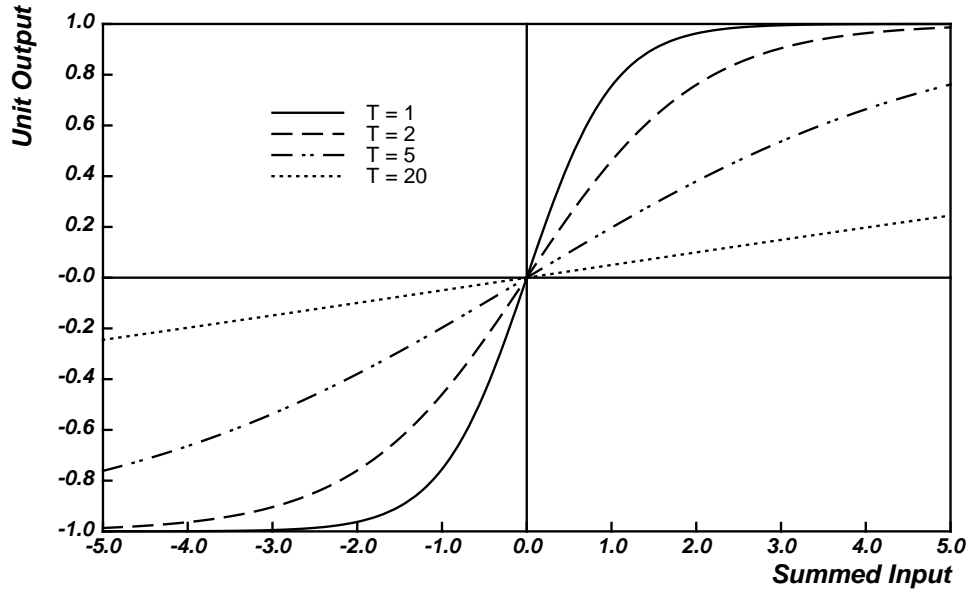


Figure 5.1: The input-output function of units in a DBM for four different temperatures.

from -1 to 1 instead of 0 to 1, and T is a parameter called “temperature” that adjusts the sharpness of the sigmoid (see Figure 5.1). Also, each connection is bi-directional and each weight is symmetric, so that $w_{ij} = w_{ji}$.

5.1.1 Energy minimization

As in a back-propagation network, input is presented to the network by clamping the states of some designated input units. If the other units in the network update their states synchronously and repeatedly according to equation 5.1, it can be shown (Hopfield, 1984) that the network will eventually settle into a set of states corresponding to a minimum of the “free energy” function,

$$F = - \sum_{i < j} s_i s_j w_{ij} + T \sum_i (s'_i \log s'_i + (1 - s'_i) \log(1 - s'_i)) \quad (5.2)$$

where $s'_i = (s_i + 1)/2$. The first term corresponds to the “energy” of the network, and measures the extent to which the states of units satisfy the constraints imposed by the weights. If two units have a positive weight between them and both have positive states (satisfying the constraint), the contribution of the weight to the energy will be positive, thus reducing the total free energy. If the units have states of opposite sign (violating the constraint of the weight), their contribution will be negative and will increase the free energy. The second term corresponds to the negative of the “entropy” of the network (weighted by temperature), and measures the degree to which unit states are at their extremes. At $T = 1$, the term for a unit has a minimum value of $\log(0.5) = -0.693$ when the unit is least extreme (has a state of 0) and approaches zero as the unit’s state approaches ± 1 . Minimizing the free energy F amounts to finding non-extreme unit states that satisfy the weight constraints.

It may help to think of a “state space” that is analogous to weight space, but has a dimension for the state of each unit in the network, and an extra dimension for free energy. For a given set of weights, each possible pattern of activity over the units can be represented as a point in state space, whose height along the extra dimension corresponds to its free energy. The entire set of these points forms an *energy surface* in state space, with hills and valleys. The initial unit states define a starting point on this surface. As each unit updates its state according to Equation 5.1, the pattern of activity of the network as a whole can be thought of as descending along the energy surface to find a minimum. This minimum is exactly what we have been calling an “attractor,” and the energy valley containing it, its “basin” of attraction.

5.1.2 Simulated annealing

The network as defined thus far will always settle into *some* minimum of the free energy function F . It is possible to help it find a *good* minimum, with a low value of F , by varying the temperature T during settling. In particular, it is useful to start T at a very high value T_{init} , corresponding to a very flat sigmoid function, and then gradually reduce it, sharpening the sigmoid, to a final value of 1. In our simulations, we use an exponential decay rate for T ,

$$T^{(t)} = 1 + T_{init}d^t \quad (5.3)$$

where $T_{init} = 50$ and $d = 0.9$. This procedure is the deterministic analogue of stochastic simulated annealing (Kirkpatrick et al., 1983), which is a commonly-used global optimization technique. It is also called “gain variation” (Hopfield & Tank, 1985; Nowlan, 1988) because the summed input of each unit is multiplied by a gain factor of $1/T^{(t)}$ that gradually increases during settling. The rationale for this procedure is that it provides a kind of progressive refinement. At high temperature, the input to a unit must be very large for it to produce any significant response (see Figure 5.1 for $T = 20$). Thus only the units that are most strongly constrained to have positive or negative states initially become active. As the temperature is lowered, units require less input to become active, and so become sensitive to weaker constraints. Only near the end of annealing do very subtle constraints have influence.

Annealing can also be understood in terms of its effect on the energy surface. A high temperature amounts to smoothing out the surface, leaving only the largest-scale hills and valleys to influence movement. As the network descends this surface to find a minimum, the concurrent reduction in temperature causes this minimum to separate into hills and valleys at a finer scale. As annealing progresses, the network descends into smaller and smaller minima, until at $T = 1$ it settles into a minimum at the finest scale, corresponding to the final set of unit states.² The states of the output units at this minimum constitute the response of the network to the input.

²To be precise, temperature smoothes out (low-pass filters) the energy surface in a space defined by the *input* to each unit, rather than its state. In state space, the actual effect of temperature is to constrain the network to remain close to the origin unless there is a strong pressure (large gradient) in some direction.

The settling process in a DBM is analogous to the forward pass in back-propagation, in the sense that both compute a set of output states for a given input. However, the existence of a well-defined energy function that characterizes this process is a major advantage of a DBM. While it is possible to compute the value of F for the states and weights in a back-propagation network, there is no direct relationship between this value and the actual operation of the network. In contrast, the value of F for a DBM, either during settling or at a minimum, provides a direct measure of how well the network is satisfying the constraints of the task. Furthermore, it is possible to compute F separately for different sets of connections and units. This will allow us to locate *where* in the network constraints are being violated when it produces an error under damage.

Another advantage of a DBM over the type of back-propagation network we have used thus far is that the settling process is much more gradual—typically involving a few hundred iterations, compared with 14 for the back-propagation networks. While this significantly increases the computational demands of simulations, it enables a much finer-grained analysis of the time-course of processing an input. For example, we can compare the “goodness” of the semantic and phonological representations (defined in terms of free energy) throughout the course of pronouncing a word. However, the need for long settling times makes the procedure somewhat less biologically plausible, since individual neurons can generate only about 100 spikes in the time required by humans to interpret visual input (Feldman & Ballard, 1982).

5.1.3 Contrastive Hebbian learning

Initially, the weights in the network are set to small random values (between ± 0.5 in our simulations). When an input is presented, the network will settle into a minimum of F , perhaps even the best possible minimum if simulated annealing is used. However, because the weights are random, the states of the output units at this minimum are very unlikely to correspond to their correct states for this input. Thus we need a procedure for adjusting the weights in the network to make it more likely that the minimum that the network settles into given some input has the appropriate output unit states.

The learning procedure for a DBM is remarkably simple and intuitive, although its derivation is beyond the scope of this thesis. It is directly analogous to the corresponding procedure for stochastic Boltzmann Machines (Ackley et al., 1985). It takes the form of two “phases” for each input: a *negative* and a *positive* phase. The negative phase is just the settling process described above: the states of the input units are clamped and the network is annealed to settle into a set of states corresponding to a free energy minimum. The positive phase is run exactly like the negative phase except that, in addition to clamping the input units, the output units are clamped into their correct states. Intuitively, the positive phase amounts to guiding the network to produce the correct response, and the negative phase amounts to letting the network try to produce the correct response on its own.

If the network has learned the task, the states of the output units should be the same in the positive and negative phases. We will use s_i^- to designate the state of unit i at the minimum for the negative phase, and s_i^+ for its state at the minimum for the positive phase. If each weight is changed according to

$$w_{ij} = \epsilon (s_i^+ s_j^+ - s_i^- s_j^-) \quad (5.4)$$

then, for small enough ϵ , the network performs steepest descent (in weight space) in an information-theoretic measure G of the difference between the output unit states in the positive and negative phases (Hinton, 1989b).³ The form of this learning rule is simply the product of unit states in the positive phase minus their product in the negative phase.⁴ This makes sense if we think of the states in the positive phase as roughly corresponding to “correct” behavior, and remember the discussion above on how states and weights contribute to the total free energy. If the states of the two units in the positive phase are either both positive or both negative, it is good (i.e. lowers the energy) for the weight to be positive, and it is incremented. We subtract off the product for the “incorrect” performance in the negative phase. If the product is not as high in this phase as in the positive phase, the net weight change will be positive. This increase in the weight will make it more likely in the future for one unit to be active when the other is active, thus increasing the product of their states. In this way, learning can be thought of as “shaping” the energy surface, lowering the surface (decreasing the energy) for good combinations of states and raising it for bad ones. These changes make it more likely that the network will settle into a good minimum on the next presentation of the input.

Contrastive Hebbian learning is more biologically plausible than back-propagation for a number of reasons. Although the procedure still requires information about the correct states of output units, this information is used in the same way as information about the input—that is, by propagating weighted unit activities, rather than passing error derivatives backward across connections. This difference makes it easier for one part of a large DBM to train another, if the first part can appropriately set the states of the output units of the second part. In addition, there is direct neurophysiological evidence for a Hebbian learning mechanism in at least some parts of the brain (Cotman & Monaghan, 1988; Lynch et al., 1984). Although the need for symmetric weights is of some concern, connection pathways between brain areas are virtually always reciprocal (Van Essen, 1985), and initially asymmetric weights gradually become symmetric if they are given a slight tendency to spontaneously decay towards zero (Galland & Hinton, 1989; Hinton, 1989b).

Although contrastive Hebbian learning in DBMs is a relatively new learning paradigm, it


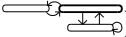
³Actually, this is only true if, in the negative phase, the probability of an output vector given an input vector is defined in terms of the free energies of the minima that the network actually settles to in the positive and negative phases, rather than by interpreting the real-valued output vector as representing a probability distribution over possible binary output vectors under a maximum entropy assumption (i.e. that the unit states represent independent probabilities).

⁴Learning rules that change the weight of a connection based on the product of the activities of the two connected units are referred to as “Hebbian” in recognition of Donald Hebb, who first proposed such a rule (Hebb, 1949). The rule is termed “contrastive” because it involves taking the *difference* to two such products.

has been applied to problems of moderate size with reasonable success (Galland & Hinton, 1990; Peterson & Hartman, 1988). In general, the number of required training presentations is comparable to that for back-propagation, although a DBM can require considerably more computation in processing each example due to its more gradual settling process.

Both back-propagation and contrastive Hebbian learning can be characterized as performing gradient descent in weight space in an explicit measure of how well the network is performing the task. This has lead most researchers to assume that the nature of the representations developed by the two procedures in most tasks would be qualitatively equivalent. However, the ways in which they compute weight derivatives based on unit states are quite different. In particular, the processing dynamics of a DBM as it settles to an attractor are much more gradual and interactive than the type of back-propagation networks we have investigated thus far. These differences raise the issue as to whether the lesion results we have obtained with back-propagation arise only in networks trained with that powerful, rather implausible procedure. In order to investigate this issue, we now define a version of the task of reading via meaning, and describe a DBM architecture for accomplishing it. After training the network with contrastive Hebbian learning, we systematically lesion it and compare its impaired performance with that of damaged back-propagation networks.

5.1.4 The task

In order to help the DBM learn the *structure* between the input and output patterns (i.e. to reproduce the co-occurrences of unit states), we will use a more “symmetric” version of the task of reading via meaning than was used with the back-propagation networks. Specifically, the network will be trained to map between orthography and phonology via semantics *in either direction*. This requirement can be broken down into three subtasks: (a) generate semantics and phonology from orthography (b) generate orthography and phonology from semantics, and (c) generate semantics and orthography from phonology. Although only the first subtask is strictly required for reading via meaning, training on the other subtasks ensures that the network learns to model orthographic structure and its relationship to semantics in the same way as for phonological structure.⁵ This is important if we want to use free energy to compare the “goodness” of each kind of representation. Also, learning the task in both directions should result in stronger and more robust attractors, in a similar way as for the back-propagation networks with feedback connections ( and )⁵. In order to make generating orthography as closely analogous as possible to generating phonology, we use the original H&S representations for letters, involving a position-specific “grapheme” unit for each possible letter in a word.

⁵Our use of a training procedure that involves learning to produce semantics from phonology in addition to producing phonology from semantics is in no way intended to imply a theoretical claim that input and output phonology are identical—it is solely a way of helping the network to learn the appropriate relationships between semantic and phonological representations.

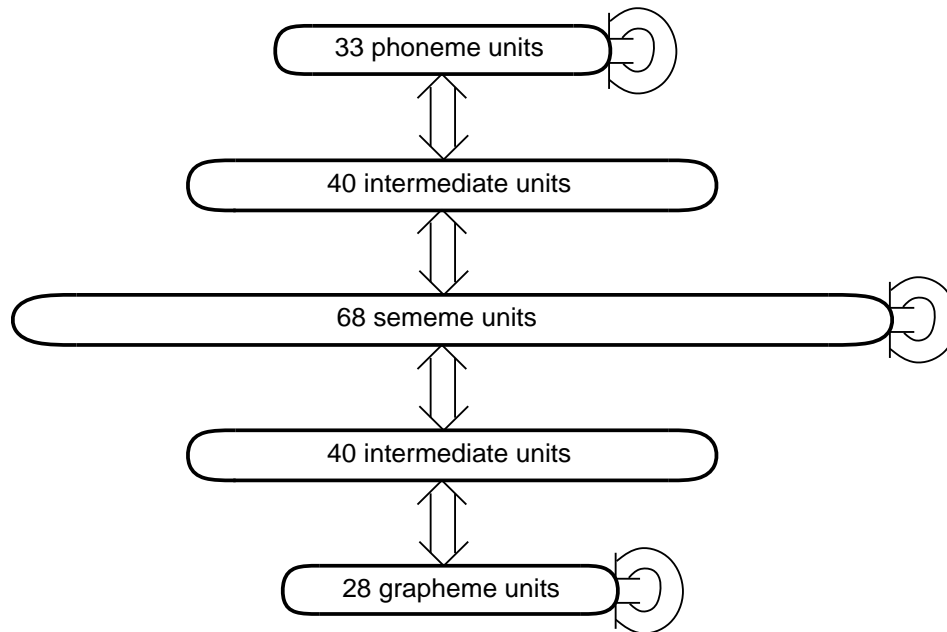


Figure 5.2: The DBM architecture for mapping among orthography, semantics, and phonology.

5.1.5 The network architecture

Figure 5.2 depicts the architecture of a DBM for mapping among the orthography, semantics, and phonology. The network has 40 intermediate units bi-directionally connected with the 28 grapheme units and 68 sememe units, and another 40 intermediate units bi-directionally connected with the sememe units and 33 phoneme units. Each of these sets of connections has full connectivity density. In addition, there is full connectivity *within* each of the grapheme, sememe, and phoneme layers, except that units are not connected with themselves. In total, the network has 11,273 bi-directional connections. This is about twice the number of connections in one of the back-propagation networks. This extra capacity is justified because contrastive Hebbian learning is not as efficient as back-propagation in using a small number of weights to solve a task.

5.1.6 The training procedure

The procedure used to train the DBM is exactly that described above, with a slight elaboration. In order to train the network to perform each of the three subtasks mentioned previously, each presentation of a word involved three negative phases. First, the grapheme units were clamped to the letters of the word, and the network was annealed to settle into states for the semantics and phonology. Second, the semantics of the word were clamped correctly, and the network generated orthographic and phonological representations. Finally, the phonemes of the word were clamped, and activity patterns over the sememe and grapheme units were computed. The pairwise products of unit states in each of these conditions are subtracted from the pending weight changes, according

to Equation 5.4. The positive phase involved clamping the grapheme, sememe, and phoneme units appropriately, and computing states for the intermediate units.⁶ In order to balance the three negative phases, the products of unit states in the positive phase are multiplied by three before being added into the pending weight changes. These pending changes are accumulated for each word in turn, at which point the weights are actually changed (using a weight step $\epsilon = 0.01$) and the procedure is repeated. After slightly more than 2100 such sweeps through the word set, the state of each grapheme, sememe, and phoneme unit was within 0.2 of its correct states during each of the three negative phases.

In order to provide a sense of the behavior of the trained network in processing a word, Figure 5.3 displays the states of the units in the network at various times during the negative phase in which the orthography of the word RAT is presented. Because temperature is very high for the first iterations, most (non-input) unit states are near zero. Gradually, units in the first intermediate layer start to become active due to direct orthographic input. By around iteration 30, this initial activity begins to generate semantic activity, which in turn generates activity in the output half of the network by iteration 35. Because only three of the 33 phoneme units should have a positive state for any given word, these units have a strong negative biases, producing negative states at iteration 40. Semantics continues to improve, although it is still far from the correct semantics for RAT, as shown by comparison with the states for that last iteration. Close inspection reveals that the erroneous semantic features are due to contamination with the features for CAT. However, even before the semantic pattern settles completely it begins to activate the appropriate phonemes—first the vowel around iteration 50, and then the consonants. Between iterations 60 and 75, the phoneme units clearly settle into the correct pronunciation. Interestingly, some semantic features are still undecided or incorrect at this state (e.g. the two leftmost features, relating to size). The correct phonology feeds back to semantics to provide additional clean-up, and by iteration 100 all of the semantic features are in their correct states. In this way, the DBM behaves quite differently from networks that map from orthography to phonology via semantics in a strictly feed-forward manner (i.e. all the back-propagation networks without feedback connections). Having learned to map between semantics and phonology in both directions, it takes advantage of their interaction to settle into the correct representations for each.

The settling behavior of the DBM when presented with other words is qualitatively similar, although there is some degree of variability in the time course of settling at the semantic and phonological layers. Most words generate correct semantics relatively quickly, but some take quite a long time to generate phonology from semantics. In a sense, this slowness in the process of generating phonology from semantics is analogous to the word-finding difficulties seen in most aphasic patients, as well as occasionally in normals. A more direct comparison of the time course

⁶No settling is required in the positive phase because all of the connections of both sets of intermediate units are from units that are clamped. In this case, the final states that these units would ultimately achieve if settling were used can be computed directly using no “sluggishness” in their update functions (i.e. $\lambda = 0$ in Equation 5.1).

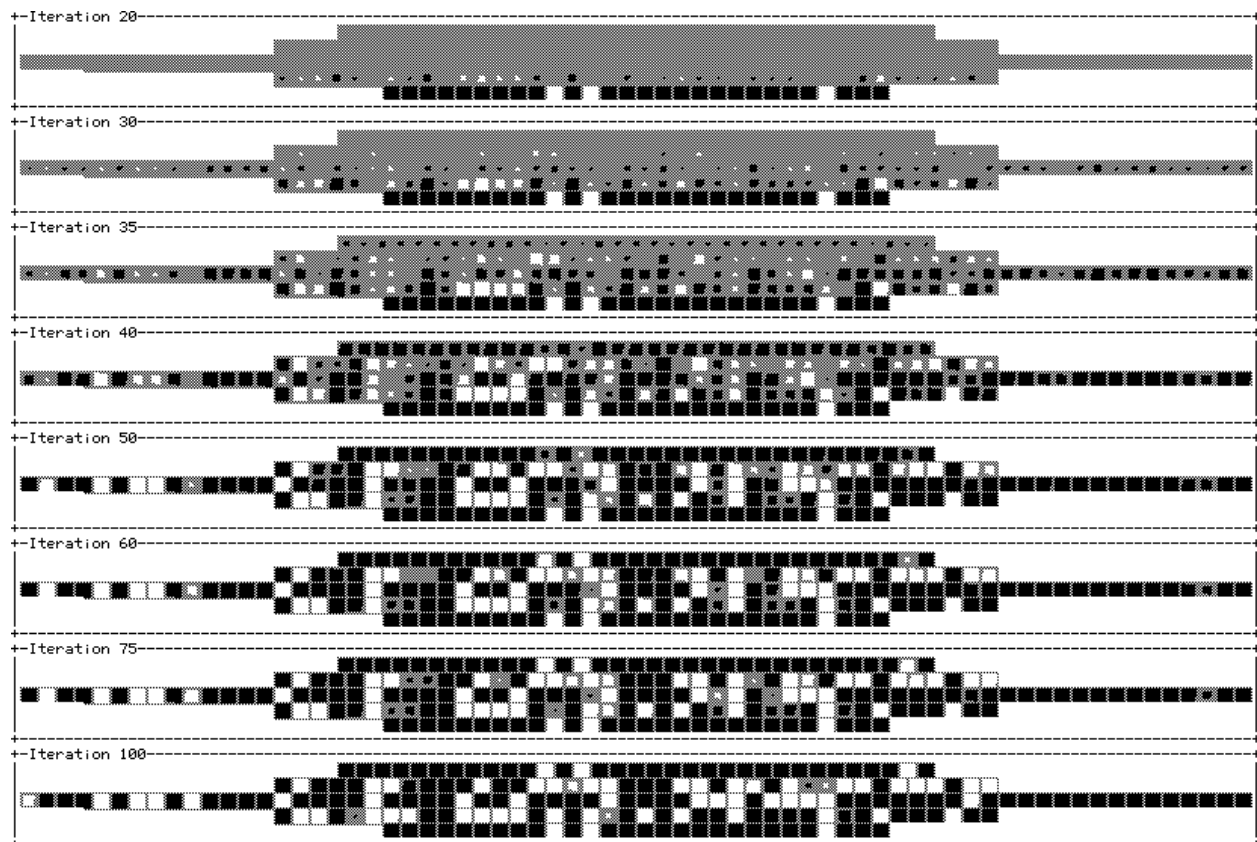


Figure 5.3: The states of the DBM at selected iterations in processing the word RAT. Each row of the display for an iteration represents a separate layer of units, with grapheme units at the bottom, sememe units in the long middle row, and phoneme units at the top. The second and fourth rows are the input and output intermediate units, respectively. The state of each unit is represented by the size of a black (for negative) or white (for positive) blob. A grey square indicates that the unit has a state near zero. Thus the bottom (orthographic) row for each iteration has three white squares, corresponding to the three graphemes of RAT that are clamped on throughout settling.

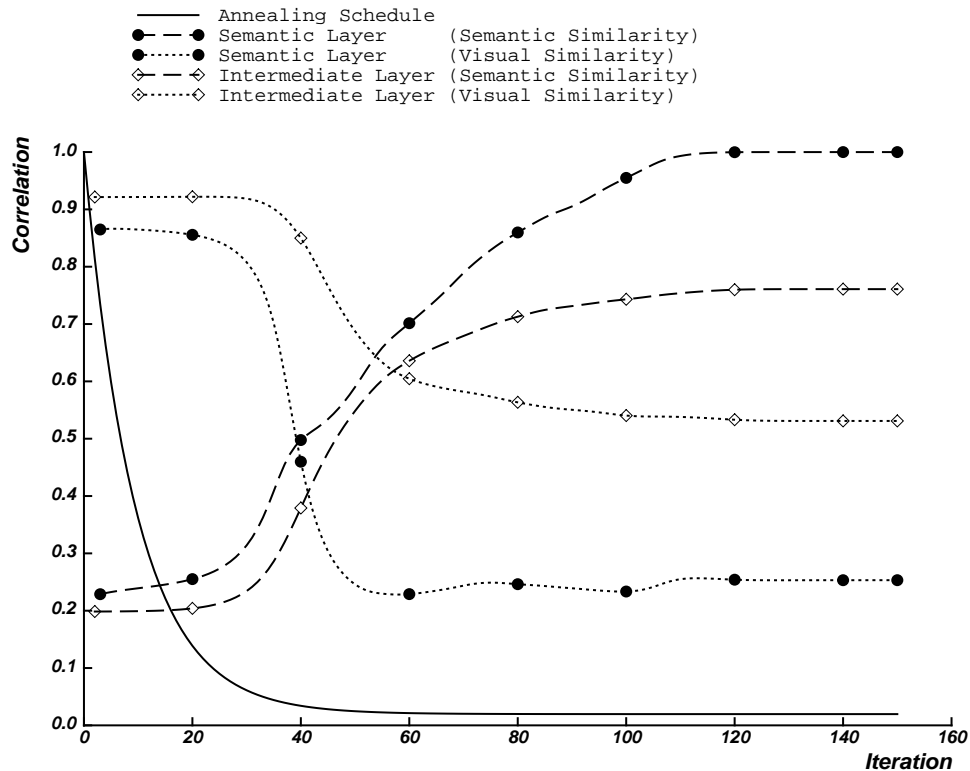
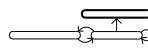


Figure 5.4: The correlation coefficients with the visual and semantic similarity matrices for the similarities among word representations at the (input) intermediate and semantic layers of the DBM for each iteration up to 150. The annealing schedule, plotted as a proportion of the initial temperature $T_{init} = 50$, is included for comparison.

over which the network solves different parts of the task is obtained by examining the changes across iterations in the correlations of the similarity matrices for word representations at a given layer with the matrices for orthographic, semantic, and phonological representations. As with the back-propagation networks, these correlations measure the degree to which representations are visually, semantically, or phonologically organized. They also allow us to compare the nature of intermediate representations developed by a DBM with those developed by back-propagation.

Considering the “input” half of the DBM first, Figure 5.4 presents the visual and semantic correlations for representations at the first intermediate layer and semantic layer of the DBM across iterations. The annealing schedule (i.e. the variation of temperature T over iterations) is included for comparison. The pattern of results is fairly similar to those for the  back-propagation network (Figure 4.9, p. 90). Representations at the two layers are rather visually organized during the initial iterations. However, the states of these units are all very near zero early on and so their similarities have little significance. Once the temperature has lowered sufficiently to allow unit states of reasonable magnitude (around iterations 30–40) the semantic layer quickly loses most of its visual organization, and begins to steadily better approximate semantic similarity.

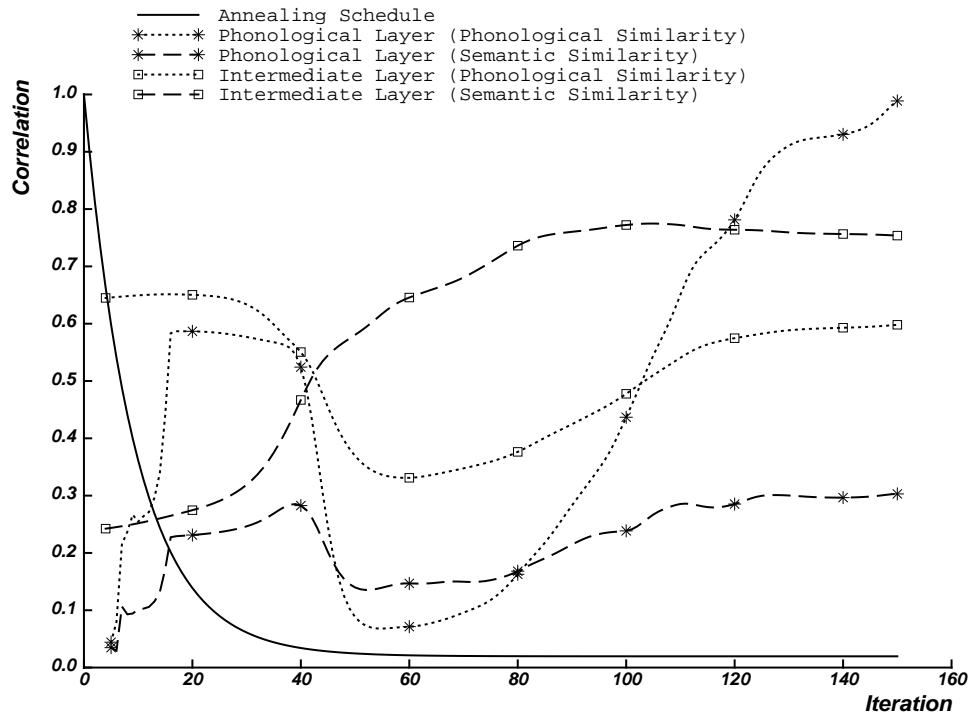


Figure 5.5: The correlation coefficients with the semantic and phonological similarity matrices for the similarities among word representations at the (output) intermediate and phonological layers of the DBM for each iteration up to 150.

The intermediate layer more gradually finds a compromise between visual and semantic similarity, eventually generating representations that are somewhat more semantically than visually organized. Interestingly, these final correlations are somewhat higher than those for the back-propagation network—the DBM has succeeded in generating representations that are *both* more visually and semantically organized than has back-propagation. In a sense, there is less “unexplained” variation in the relationships among representations, and they are more appropriate to mediate in mapping between orthography and semantics.

The correlations for the output intermediate layer and phonological layer, shown in Figure 5.5, are even more interesting. For these layers it seems more appropriate to consider phonological rather than visual similarity, although the pairwise similarities of these two representations are highly correlated (0.87). Considering the intermediate layer first, once temperature has dropped sufficiently the representations at this layer become increasingly semantically organized. Their phonological similarity drops at this point, but then gradually increases as the representations at the phonological layer improve. Presumably the increase in phonological similarity at the intermediate layer reflects more coherent feedback from the phonological layer, of the sort evidenced in the display of unit states over iterations in processing the word RAT. The phonological layer never shows any strong semantic influences in the organization of its representations, but the pattern of phonological correlation over iterations is rather peculiar. Ignoring the quick initial rise for

the moment, when the temperate has dropped the similarities among representations become very unlike their final similarities. This drop in correlation reflects the fact that unit states at this point (around iterations 50–60) are driven more by their strong negative bias than by influences from semantics. As shown in Figure 5.6, the phonological patterns for many words are rather uniformly negative, and so their pairwise similarities are all high, unlike their final similarities. Gradually, as input from semantics causes particular phoneme units to adopt positive states, the similarities among representations better approximate a phonological organization (see Figure 5.7). However, even at iteration 120 there are still some words (e.g. PORE and LIME that have yet to settle into a clean pronunciation.

Although the similarities among phonological representations changes quite dramatically across iterations, the change in the degree to which these representations satisfy the constraints imposed by the weights is much more well-behaved. Figure 5.8 displays the energy at the semantic and phonological layers, as well as the total energy of the network, averaged across connections to normalize the comparison. The network as a whole shows a smooth descent in energy, with the largest change occurring between iterations 30 to 50 when the temperature lowers below 3.0. The semantic and phonological layers behave similarly, with the latter layer showing the sharpest drop at this point. Interestingly, this transition corresponds to the point at which phonological similarities become *least* like their final similarities. Uniform negative activity at the phonological level satisfies almost all of the phonological constraints—only a slight reduction in energy results from activating the three correct phonemes for the word over the next 100 iterations.

In comparing the training and operation of the DBM with that of the back-propagation networks, it is important to keep in mind that processing one word in the DBM requires about 40 times more computation.⁷ On the other hand, the DBM has the significant advantage that it was trained all at once—back-propagation networks had to be trained incrementally, using a rather *ad hoc* procedure in the case of the output networks (see Section 3.2.2). In addition, the DBM is performing a more complex task by learning to map between orthography and phonology in either direction. However, our major interest is to compare the effects of damage on behavior of these two types of network in reading via meaning rather than the time required to learn the task *per se*.

5.1.7 The effects of lesions

After training, each of the sets of connections in the DBM were subjected to 20 instances of lesions over the standard range of severity. Since we are primarily concerned with the task of generating semantics and phonology from orthography, we only considered behavior in the negative phase in

⁷We can approximate the computational demands of presenting a word during learning by the number of connections \times the number of phases \times the number of iterations per phase. The DBM has about twice the number of connections and requires four phases, compared with two for a back-propagation network (the forward and backward passes). In addition, the DBM requires about 10 times more iterations to settle (about 150 vs. 14 for one of the back-propagation networks).

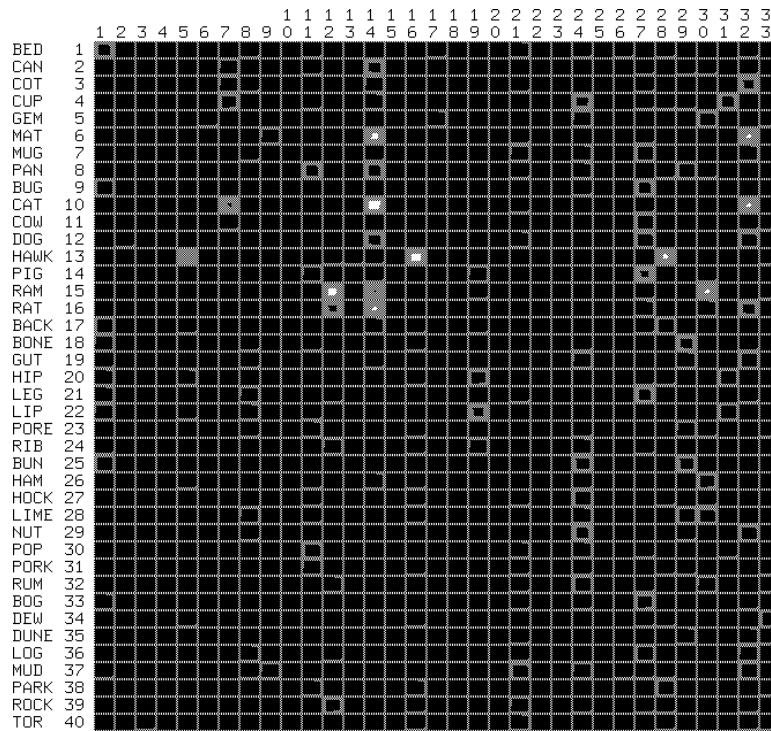


Figure 5.6: The phonological representations of words at iteration 50 in the DBM.

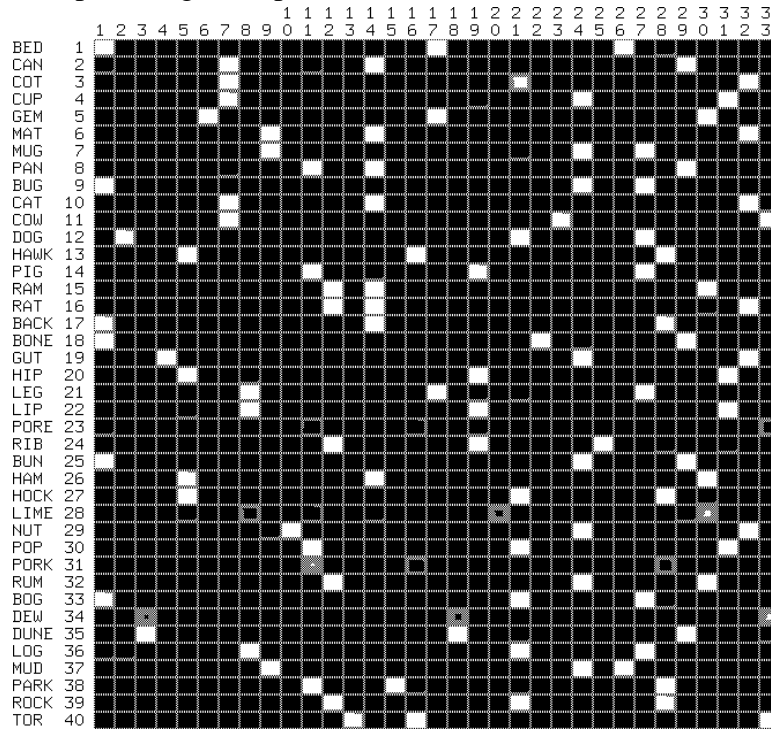


Figure 5.7: The phonological representations of words at iteration 120 in the DBM.

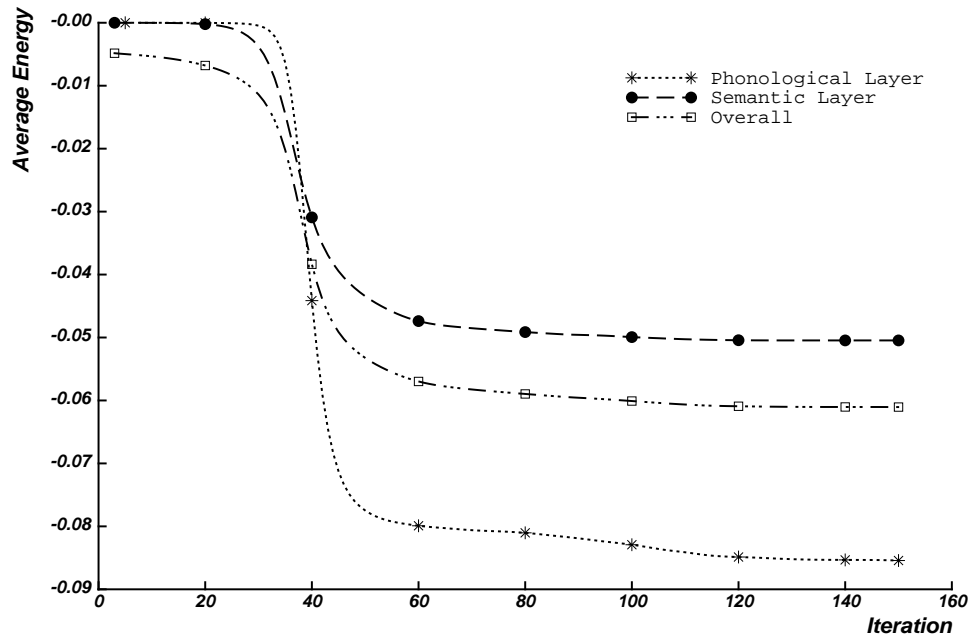


Figure 5.8: The energy of the semantic layer, phonological layer, and the complete network, averaged over number of connections.

which the grapheme units are clamped. For each lesion, correct, omission, and error response were accumulated according to the same criteria as used for the back-propagation networks. Figure 5.9 presents the overall correct rates of performance of the DBM, for both “input” and “output” lesions. As a comparison, consider the correct performance data for the corresponding lesions to the replication of the H&S network (Figures 3.7 and 3.11, pp. 60 and 65).

This network is similar to the DBM in that it (a) uses the same orthographic representations, (b) was not trained with noisy input, (c) has intra-sememe connections. Considering “input” lesions first, the DBM is more robust to $I \leftrightarrow S$ lesions, while $G \leftrightarrow I$ lesions are equally debilitating in the two networks. A comparison of “clean-up” lesions is complicated by the differences in architecture—the H&S network has a clean-up pathway, while the DBM has only intra-sememe connections. In general, the DBM appears to be slightly more resistant to $S \leftrightarrow S$ lesions than the H&S network is to either $S \Rightarrow C$ or $C \Rightarrow S$ lesions, except perhaps when using the IP output network. As for output lesions, the DBM is somewhat more robust to lesions of the direct pathway than is the H&S network. However, clean-up lesions in the two networks result in similar behavior, producing a sharp decline in correct performance with increasing lesion severity.

An interesting characteristic of the DBM is that it tends to settle into unit states that are very close to ± 1 , even under damage. This results in very clean phonological output when it responds, as can be seen in the distribution of the minimum probability of the output vector producing each phoneme, for correct responses, errors and omissions (see Figure 5.10 and compare with Figure 3.3, p. 54, presenting data for the replication of the H&S back-propagation network). In the DBM, the

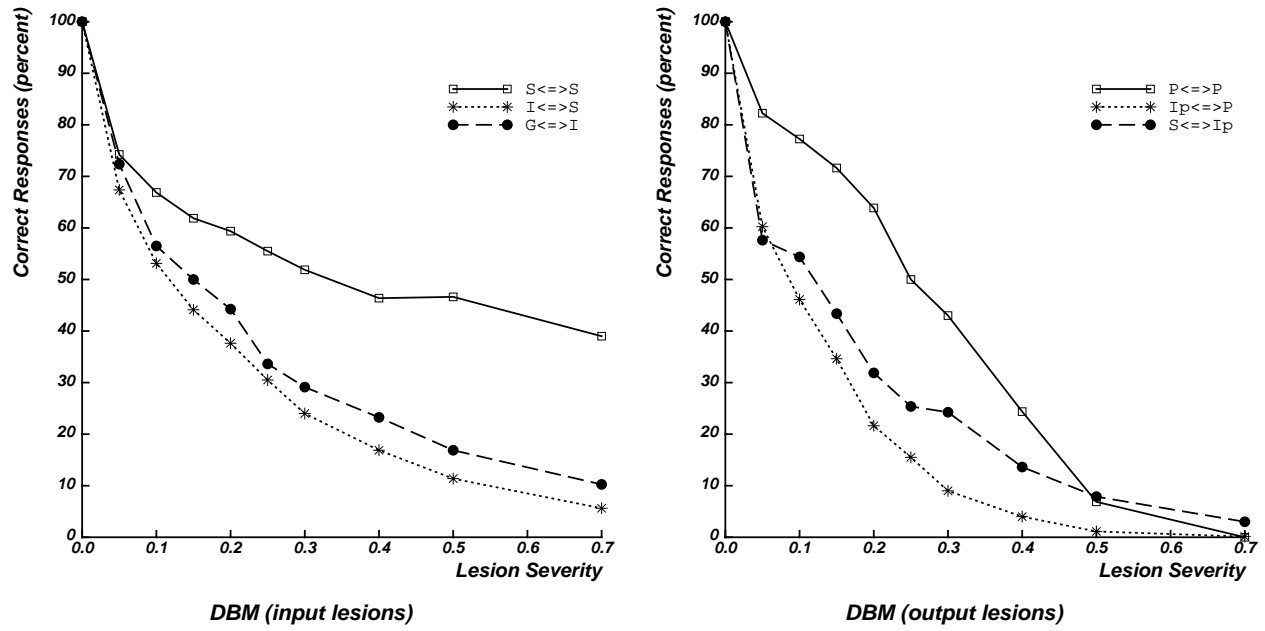


Figure 5.9: Overall correct performance of the DBM after lesions to the “input” connections (left) and “output” connections (right).

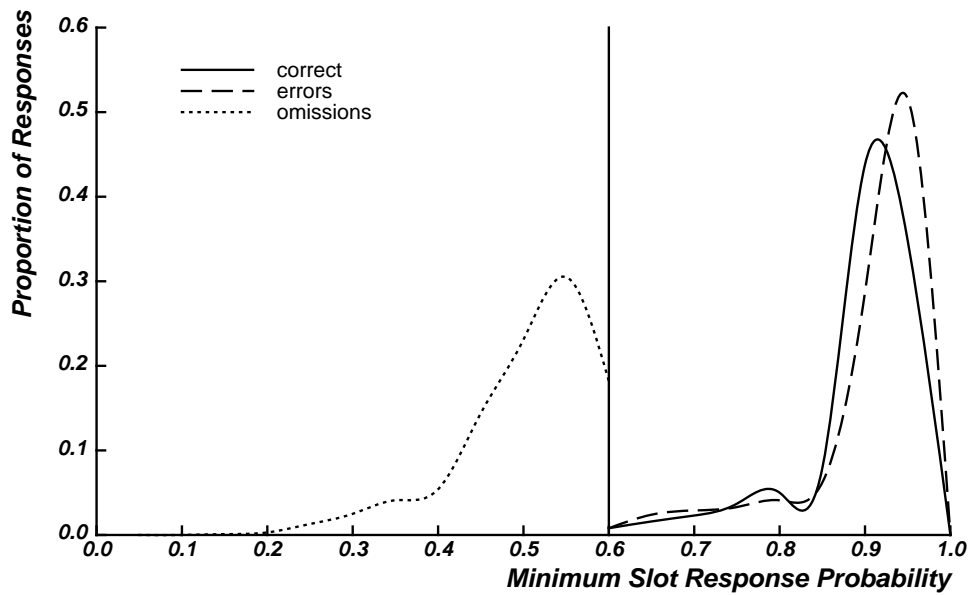


Figure 5.10: Distribution of the minimum output probability at any slot for correct responses, errors, and omissions.

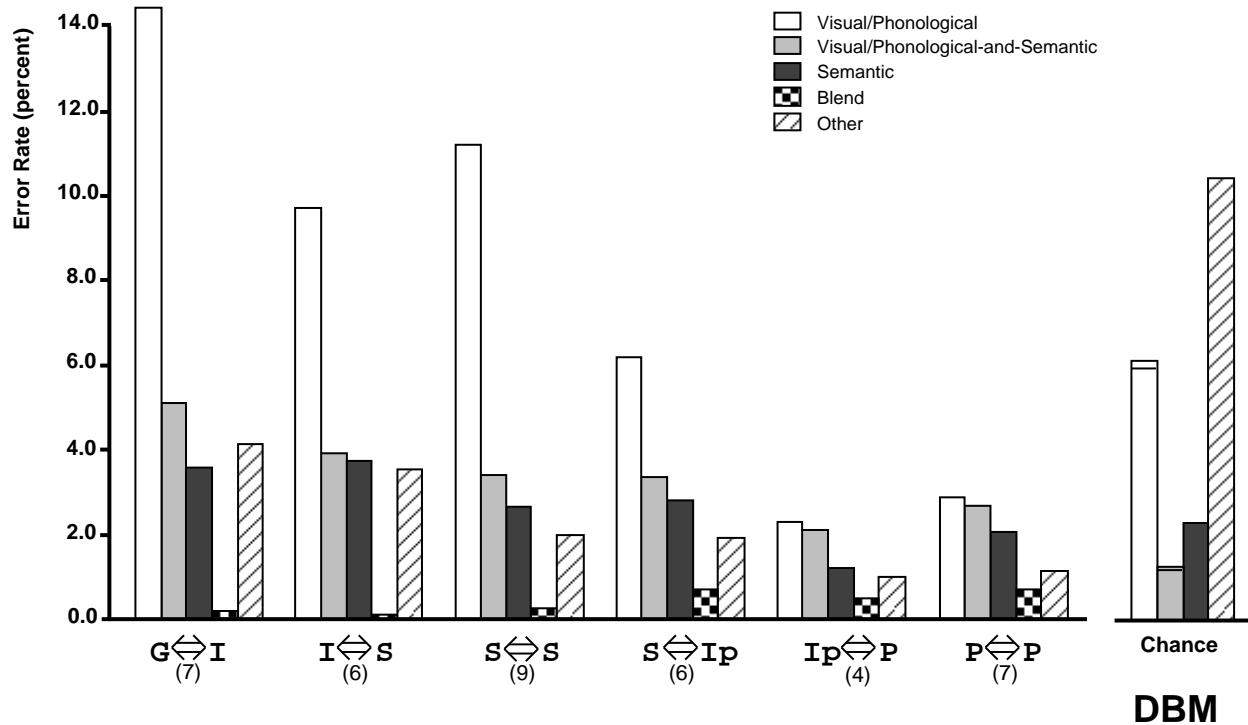
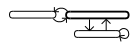


Figure 5.11: Error rates produced by lesions to each main set of connections in the DBM.

worst phoneme has a probability above 0.8 for almost all correct and omission responses, while very few are above this level for the back-propagation network. In addition, the large majority (90.8%) of omissions fail the requirement that exactly one phoneme be active—no phoneme is active in 87.2% of these. Only 9.2% of omissions fail because of the criterion of a minimum slot response probability of 0.6 for responses. Thus in the DBM this criterion could be eliminated entirely without substantially altering the results.

Figure 5.11 presents the distribution of error types for each lesion location of the DBM, averaged over severities that resulted in correct performance between 20–80%. Comparing with results for input lesions to the H&S network (Figures 3.8, p. 62), the DBM is producing much higher error rates—about 4–8 times the rates using the noIP output network, and about twice the rates using the IP network. In fact, the distribution of error types is quite similar for the latter network and the DBM. Both show a very high proportion of visual errors for lesions to input pathways. Furthermore, like the replication of the H&S network with an output system, the DBM shows very low rates of blend responses. This is interesting because, unlike in the development of the noIP and IP output networks, no special effort was made to prevent blends in the design or training of the DBM. Their absence appears to be a natural and encouraging consequence of the nature of the attractors developed by the DBM.

The pattern of error rates for output lesions to the DBM is quite different from that for the replication of the H&S network with an output system (Figure 3.12, p. 67). The error rates for

lesions to the direct pathway of the DBM ($S \leftrightarrow I_p$ and $I_p \leftrightarrow P$) are lower, and less biased towards visual errors. In addition, the DBM produces far fewer “other” errors than either back-propagation output network. Perhaps more striking, phonological clean-up lesions in the DBM ($P \leftrightarrow P$) still produce significant error rates, fairly evenly distributed across type, while the analogous lesions in the back-propagation networks ($P \Rightarrow C_p$ and $C_p \Rightarrow P$) produce virtually no error responses. With phonological clean-up damage, the DBM can use the bi-directional interactions with the intermediate units as a residual source of clean-up. This redundancy of clean-up is similar to that of the hybrid  network (see Section 4.4.5).

All lesion locations in the DBM show a mixture of error types, and their ratios with the “other” error rates are higher than for randomly chosen error responses. In addition, the rates of mixed visual-and-semantic errors are higher for all lesion locations that expected from the independent rates of visual errors and semantic errors (although just slightly for $S \leftrightarrow S$ lesions). Thus the DBM replicates the main H&S results.

The similarity of the results produced by input lesions to the DBM with those produced by the H&S network using the IP output network lends credence to the notion that the *strength* of the attractors for words is a much more important factor in determining the pattern of results than is the procedure by which those attractors are developed. However, learning in the DBM develops strong attractors naturally, without the need for incremental training with noisy input. Furthermore, the interactive nature of processing in the DBM makes a large difference for lesions at the phonological level. Unlike the back-propagation output networks, the DBM can fall back on bi-directional interactions with semantic (via the intermediate units) to provide clean-up that can partially compensate for lesions to intra-phoneme connections.

5.2 GRAIN networks

The effectiveness of noise in facilitating the development of strong attractors in the back-propagation networks suggests that it might have further benefits within the DBM framework. In fact, a DBM can be viewed as a mean-field approximation to a standard stochastic Boltzmann Machine (SBM), in which the real-valued state of each unit in the DBM represents the expected value of the stochastic binary state of a unit in the SBM. The importance of noise suggests that an SBM implementation of the task of reading-via-meaning might have interesting properties. However, learning in an SBM is impractically slow because the products of binary states must be sampled over a large number of iterations in order to estimate their expected values. In contrast, the real-valued products in a DBM can be computed directly. McClelland (1990; 1991) has recently developed a stochastic elaboration of DBMs, called GRAIN networks (for Gradual, Random, Adaptive, and Interactive), that use real-valued stochastic units.⁸ To investigate the impact of noise on the

⁸Actually, GRAIN networks were developed as an elaboration of the Interactive Activation framework (McClelland & Rumelhart, 1981; Rumelhart & McClelland, 1982) in response to the need for intrinsic variability as reflected by

attractors in a DBM, we develop a GRAIN network analogous to the DBM, and compare their behavior under damage.

The principles of GRAIN networks can be embodied in a wide range of specific network formalisms—we chose a version that is as close to a DBM as possible. In particular, the type of GRAIN network we will investigate is identical to a DBM, except that there is intrinsic variability in the input to each unit at every time step. Thus the output of unit i at time t becomes

$$s_i^{(t)} = \lambda s_i^{(t-1)} + (1 - \lambda) \tanh \left(\frac{1}{T} \left(\nu_\sigma + \sum_j s_j^{(t-1)} w_{ij} \right) \right) \quad (5.5)$$

where ν_σ is noise sampled from a gaussian distribution with mean 0 and standard deviation σ (cf. Equation 5.1). We used $\sigma = 0.1$ in our simulations. In contrast to using noisy input patterns, this type of noise applies to every unit in the network throughout settling. Thus the influence of noise is more widespread in a GRAIN network than in the back-propagation networks trained with noise.

The impact of noise in a GRAIN network can be understood in terms of a constraint on the nature of the energy surface it must construct during learning. Consider the movement of the network along the energy surface as it settles in response to some input. Because of the intrinsic variability of the input (and hence output) of units, the direction that the network as a whole moves on the surface is not the exact direction dictated by the previous unit states (as in a DBM), but this direction corrupted by gaussian noise. In this way the movement of the network as it minimizes free energy resembles a “random walk” process. Nonetheless, in order for the network to correctly perform the task it must reliably reach the same (correct) minimum for each presentation of the input. Thus it must learn to shape the energy surface so as to be robust against noisy variation in the direction of descent. This can be accomplished by making the surface smooth, the basins of attraction for training patterns large, and the basins for “spurious” attractors as few and small as possible. In contrast, it is possible for a DBM to construct a very irregular energy surface and yet still perform a task correctly. As long as each particular input pattern settles into the correct minimum, the network need not “explore” other parts of state space, and so the energy surface in these regions can be arbitrarily ill-formed.

While the degree to which the energy surface is well-formed may not affect normal performance, it certainly affects the behavior of the network under damage. A lesion to connections alters the way that units interact, and so distorts the energy surface for a given input. In particular, because the contribution of each weight to the energy is additive, the energy surface consists of the sum of the surfaces for each individual weight.⁹ The surface for each weight w_{ij} is three-dimensional and shaped like a saddle, with height $s_i s_j w_{ij}$ at point (s_i, s_j) (see Figure 5.12). Diagonally opposite

empirical limitations of the original model (Massaro, 1988). However, the processing dynamics in a DBM are a special case of those in the IA model.

⁹The free energy surface on which the network performs gradient descent is this energy surface with the entropy terms of the free energy function subtracted from it. The effect of the entropy terms can be thought of as pulling up on every corner of the energy surface.

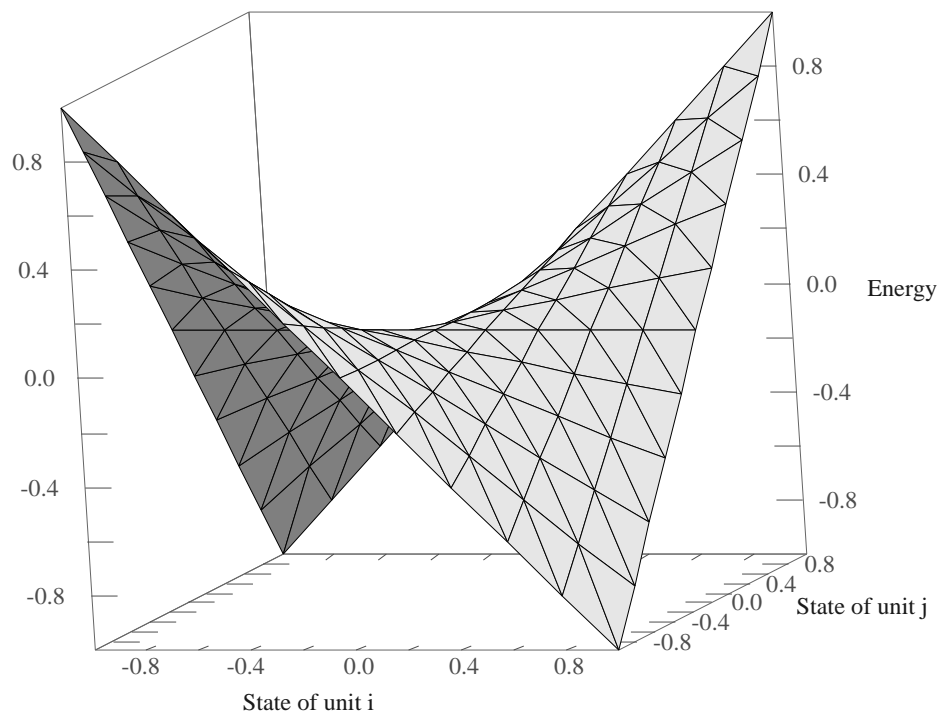


Figure 5.12: The energy contribution to total free energy of a single weight $w_{ij} = -1$ as a function of the states of units i and j . Notice that the energy is low whenever the states of i and j have opposite sign, and high when they have the same sign.

corners are either both high or both low, depending on the sign of the weight—the magnitude of the weight simply scales the difference between these high and low points. The figure displays the surface for a weight of -1 . Lesioning the connection turns this “saddle” into a flat plane with zero energy. This distorts the free energy surface for the whole network by adding in the oppositely-shaped saddle surface. The distortion produced by lesioning many connections is simply the surface which is the sum of the distortions for each individual lesioned connection. Adding this surface to the original produces the actual energy surface along which the lesioned network descends. If the original surface is smooth and has large attractor basins, gradient descent along this distorted surface is more likely to end up in the appropriate minimum. For this reason, we might expect the attractors in a GRAIN network to be more robust to damage, producing more correct as well as error responses, than in a DBM.

5.2.1 The training procedure

We developed a GRAIN network with the same architecture as the DBM (shown in Figure 5.2, p. 129) and trained it using contrastive Hebbian learning, using the same annealing schedule and learning parameters. Because the units in a GRAIN network are stochastic, the units never completely reach a fixedpoint in state space, but randomly fluctuate around it. The most general learning procedure would be to sample the correlations of unit states over a number of iterations and use their expected values in the weight update rule, just as in an SBM. However, if the amount of noise is small relative to the weights, the network will almost never jump out of a minimum as a result of the noise alone. In this case, all of the variation in unit states is caused by independent noise with zero mean, and so the expected value of the product of two unit states is the product of the states the units would have without noise.¹⁰ For this reason, the final unit states at the end of settling are computed without noise before being used in the weight update rule.

After 3500 sweeps through the training set, the GRAIN network could reliably generate any two of the orthography, semantics, or phonology of a word when given the third. The network took about a third again as long to train as the DBM due to the more difficult constraints on the energy surface needed to solve the task.

5.2.2 The effects of lesions

The GRAIN network was subjected to the same set of lesions as the DBM network, and correct, omission, and error responses were accumulated. The input to units remained noisy during the gathering of data on impaired performance. Figure 5.13 presents the network’s overall correct rates of performance. Comparing with the results for the DBM (Figure 5.9, p. 137), the rates of correct

¹⁰Fluctuations in the states of two connected units due to noise will tend to be slightly correlated due to the weight between them, so that the product of their states without noise only approximates the expected value of their product with noise.

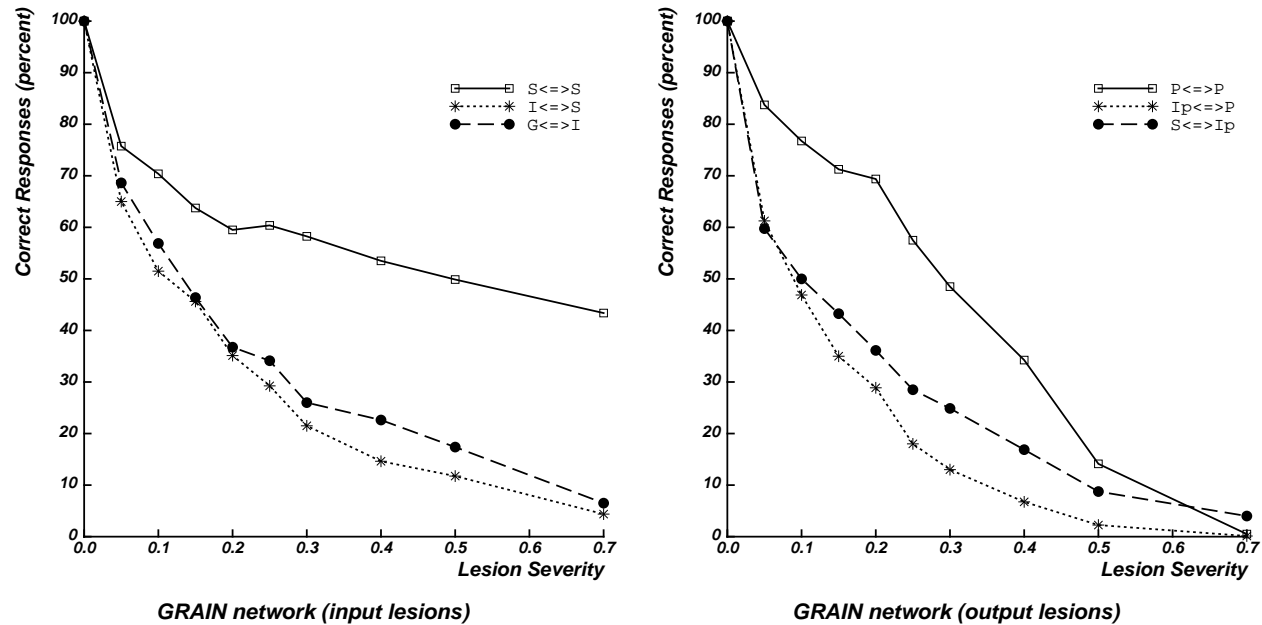


Figure 5.13: Overall correct performance of the GRAIN network after lesions to the “input” connections (left) and “output” connections.

performance of the two network are almost indistinguishable. Contrary to intuitions, the GRAIN network does not appear to be more robust to lesions than the DBM.

Figure 5.14 presents the distribution of error types for each lesion location of the GRAIN network, averaged over severities that resulted in correct performance between 20–80%. The pattern of errors is also quite similar to that of the DBM (Figure 5.11, p. 138). The major difference is that the GRAIN network has significantly higher rates of semantic errors than the DBM for almost all lesion locations. This makes sense in the following way. The amount of variation in input due to noise that a unit experiences increases as a function of its number of connections. Consider the input a unit j receives along a connection from unit i . Because the input to unit i has noise with zero mean added to it, its input to j can be thought of as a random variable with mean equal to what $s_i w_{ij}$ would be without noise (call it $s'_i w_{ij}$) and some variance dependent on the amount of noise. The summed input to j (before noise is added) is thus the sum of samples of a set of random variables. This sum is also a random variable, with mean equal to the sum of the means of the variables (i.e. $\sum_i s'_i w_{ij}$), and variance equal to the sum of their variances. Thus the mean of the summed input to a unit correctly approximates the true mean in a noiseless network, but the variance increases linearly with its number of connections. In the GRAIN network, semantic units have far more connections (149) than intermediate units (102) or phonological units (74), and so they are more drastically affected by the intrinsic noise in the states of other units. They must interact more effectively to compensate for this variability, resulting in stronger attractors at this level, and thus more semantic errors under damage.

Nonetheless, it is surprising that the GRAIN network and the DBM are so similar terms of the

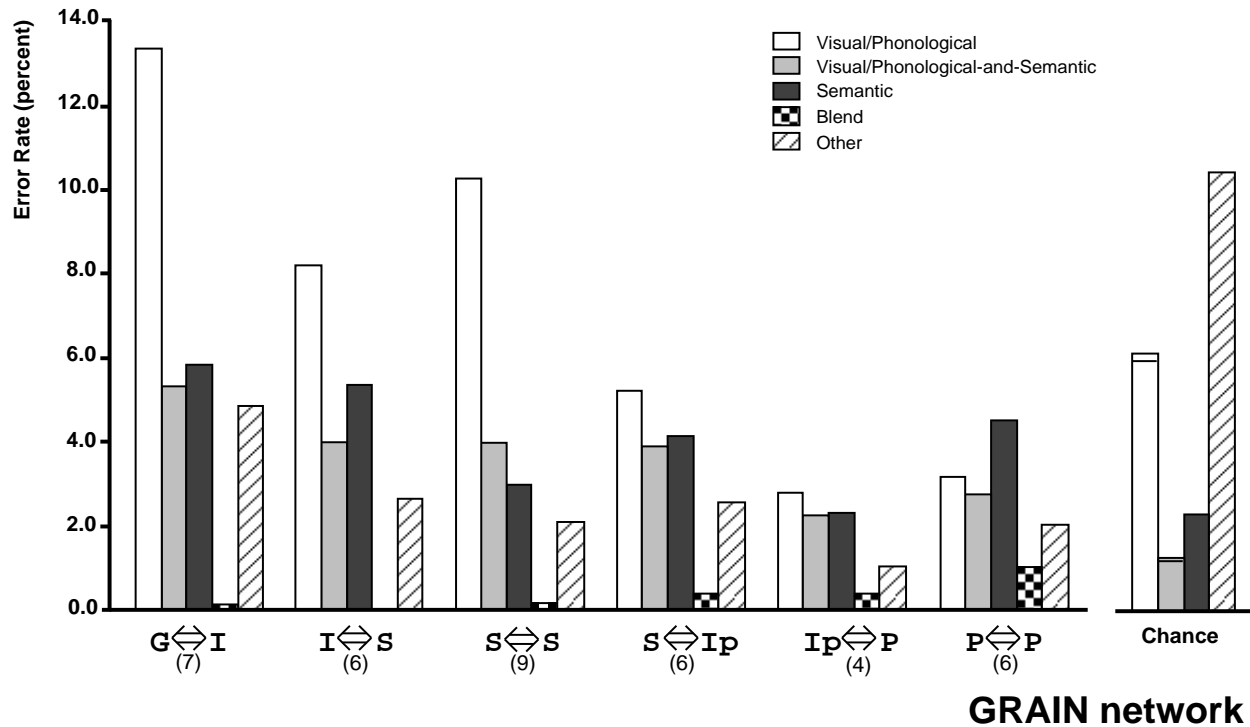


Figure 5.14: Error rates produced by lesions to each main set of connections in the GRAIN network.

nature of the attractors they develop, as reflected in their behavior under damage. One explanation may come from the behavior of the DBM during learning. The mathematical justification for the learning procedure (Hinton, 1989b) assumes that only rarely will small changes to the weights cause the network to settle into a different minimum. However, in practice this appears to be more the rule than the exception. As the weights slowly change, the network samples among a large number of minima during the negative phase(s), raising their energy to the degree to which they differ from the minima of each corresponding positive phase. As the network improves on the task, fewer and fewer of these minima remain sufficiently good for the network to settle into them. Eventually the network consistently reaches the single minimum that is most similar to the positive phase minimum, and reduces the difference until the training criteria are met. This type of variability *over weight changes* in settling to minima appears to have similar effects as the variability of unit states during a single settling in a GRAIN network. Both processes force the network to explore, and hence shape appropriately, a much larger amount of the energy surface in state space than will ultimately be traversed when the network has learned. Hence, one possible explanation for why the GRAIN network is no more robust to damage than the DBM is that in both networks the attractors have been strengthened by pressure from variability, albeit from different sources.

Both the DBM and GRAIN network serve to validate the claim that the nature of the attractors developed using back-propagation have properties under damage that are similar to those devel-

oped using alternative, more biologically plausible formalisms. But beyond plausibility, these formalisms have interesting computational characteristics that have enabled a more detailed and principled analysis of their behavior during operation. In addition, there are some aspects of the reading behavior of deep dyslexics that are much more effectively addressed using networks that have a well-defined measure of the goodness of representations. Two examples of this are the differences that some patients show in the relative confidence they have in some types of error responses, and the relative preservation of ability to distinguish words from non-words in the face of poor overt reading performance.

5.3 Confidence in visual vs. semantic errors

Patterson (1978) found that deep dyslexics D.E. and P.W. were more confident that their visual error responses were correct as compared with their semantic error responses. It is difficult to interpret these results because it is hard to know how to operationalize the notion of “confidence” in a response. One possible interpretation is that a lack of confidence arises when the system takes a long time to settle, or settles into relatively poor representations.

A network has settled to a fixedpoint for some input when the states of all the units stop changing. However, because of the nature of the energy surface, movement towards the exact fixedpoint slows down exponentially near the end of settling (Zak, 1989). This is not a problem in practice because we need only reach the fixedpoint approximately for the learning procedure to work. In the DBM and GRAIN simulations, we considered the network to have settled when the maximum change of any unit state became less than 0.01.¹¹

Figure 5.15 presents distributions of the number of iterations required to settle for correct responses, omissions, visual errors, and semantic errors produced by lesions to the DBM that resulted in correct performance between 20–80%. Not surprisingly, word presentations producing correct responses tend to settle most quickly. What is surprising is that the network takes longer on average to settle into an error response than an omission. However, remember that over 90% of omissions arise because no phoneme is active in some slot. Apparently the network is quick to turn off all the phoneme units in a slot if none of them receive sufficient support from the intermediate units as a result of damage. Accumulating enough support to fully activate a phoneme unit in each slot (and inhibit all others) often requires many more iterations. The two error types also show the most variability in settling time. While there is a high degree of overlap between the two distributions, on average visual errors settle more quickly (mean 127.2 iterations) than semantic errors (mean 139.4 iterations, $F(1, 4458) = 56.8$, $p < .001$). Thus, increased settling

¹¹In the GRAIN network, the amount of noise was lowered at the very end of annealing in order to enable units to meet the settling criterion. Under the assumption that the amount of noise is insufficient to move the network to a different attractor, determining the fixedpoint by lowering the noise is equivalent to estimating the expected values of unit states by sampling over many iterations.

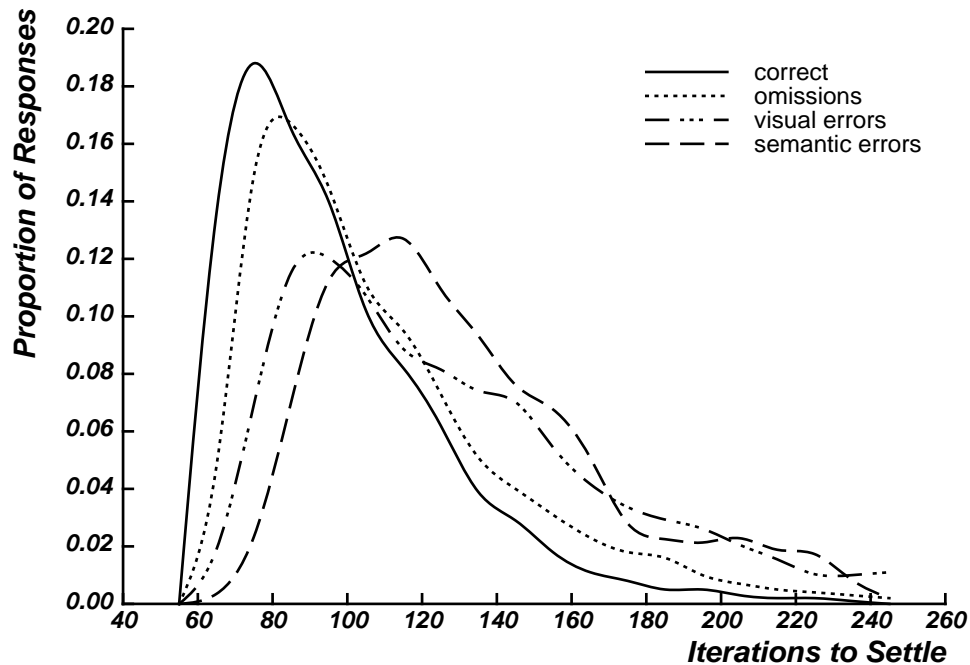


Figure 5.15: Distributions of the number of iterations to settle for correct responses, omissions, visual errors, and semantic errors produced by the DBM under damage.

time for semantic errors might account for patients' reduced confidence that these error responses are correct.

Another possible contribution to the confidence that patients have in their responses is the degree to which the system settles into “good” representations, defined to be those with low energy. We compared visual and semantic errors in terms of their energy in different parts of the network. Considering the energy in the sets of connections between semantics and phonology ($S \Leftrightarrow I_p$ and $I_p \Leftrightarrow P$), visual errors have lower energy than semantic errors in both the DBM (means -214.2 visual vs. -211.6 semantic, $F(1, 3456) = 25.0, p < .001$) and the GRAIN network (means -252.0 visual vs. -247.3 semantic, $F(1, 3839) = 101.7, p < .001$). This was true both for input and output lesions. For the sets of connections between orthography and semantics, this was true only for input lesions to the GRAIN network (means -237.0 visual vs. -234.9 semantic, $F(1, 2878) = 8.4, p < .005$)—there was no difference between the energy for visual vs. semantic errors in the DBM network ($F(1, 2647) = 1.4$). In fact, the opposite effect was true for output lesions of both networks: semantic errors have lower energy between orthography and semantics than do visual errors (DBM: means -226.9 visual vs. -232.1 semantic, $F(1, 807) = 24.2, p < .001$; GRAIN: means -246.8 visual vs. -257.8 semantic, $F(1, 959) = 166.7, p < .001$). Thus differences in energy can only account for the increased confidence that some deep dyslexics have in visual as compared with semantic errors under the assumption that their judgment is based on the energy between semantics and phonology.

Non-word Stimuli							
Close				Distant			
BUD	GEG	LIM	PIP	BERK	GAG	LUR	PET
BUT	GIM	MED	POCK	BIT	GAP	MOB	PICK
CAR	HACK	MUT	RAB	CICE	HUB	MOM	REN
DEN	HARK	NAT	ROR	DAP	HUR	NOD	RUNK
DONE	LIB	NUG	TOP	DIT	LAD	NOM	TAG

Table 5.1: The “non-words” used in the lexical decision experiment.

5.4 Lexical decision

Even when deep dyslexics are unable to read words, they can often reliably distinguish them from orthographically regular non-words. Coltheart (1980a) lists nine of the 11 cases of deep dyslexia for which there was data as being “surprisingly good” at lexical decision. For example, Patterson (1979) found that both D.E. and P.W. were near perfect at distinguishing function words from non-words that differed in a single letter (e.g. WHEN, WHIN), whereas explicit correct reading performance on the words was only 38% for D.E. and 8% for P.W. In a more difficult test involving 150 abstract words, again paired with non-words differing by a single letter (e.g. ORIGINATE, ORIGILATE), D.E. produced a d' score of 1.74; $d' = 2.48$ for P.W.¹² By comparison, $d' = 3.30$ for normal age-matched controls. D.E. read only 19 of the 150 words correctly (12.7%), while P.W. read only 31 (20.7%). Thus P.W. shows almost normal lexical decision performance with words he has difficulty reading; D.E.’s performance is significantly impaired but still much better than chance ($d' = 0$).

Hinton & Shallice (1989) attempted to model preserved lexical decision under condition of poor explicit reading performance in the following way. They constructed two sets of “non-word” stimuli with equivalent orthographic structure to the words (see Table 5.1). The non-words in the “close” set were created by changing a single letter of one of the words; those in the “distant” set differed from every word by at least two letters. The two sets are matched in the frequency with which particular letters occur at particular positions, but not with respect to the word set. These stimuli are “non-words” in the sense that they are unfamiliar to the network—it has not learned to associate them with any semantics. The fact that many of them are in fact English words (e.g. PICK) is irrelevant to the network’s behavior.

H&S modeled the task of lexical decision by changing the criteria used to generate responses. Specifically, a stimulus was accepted as a word if the proximity of the generated semantics to the nearest familiar semantics exceeded 0.7, ignoring the gap between this and other matches. The

¹² d' is a statistic from signal detection theory that measures the discriminability of two distributions in standard deviation units, independent of response bias.

rationale for using a reduced proximity criterion and no gap criterion is that the semantic match required to indicate that the stimulus is a word needn't be as precise as the match required to specify a particular word for explicit naming. However, when this procedure was applied to the responses generated by the network after damage, there was little difference between words and non-words. For example, for a lesion of $G \Rightarrow I(0.4)$, which produces 18% explicit correct performance, 67.3% of words were accepted, while 55.5% of close non-words and 64.0% of distant non-words were incorrectly accepted as words ($d' = 0.31$ and 0.09 , respectively). For $I \Rightarrow S(0.2)$ lesions (21.5% correct performance), 57% of words, 39% of close non-words, and 45% of distant non-words were accepted as words ($d' = 0.46$ and 0.30 , respectively). Thus, H&S failed to demonstrate preserved lexical decision performance in their network when explicit correct performance is poor.

In the context of modeling the non-semantic route from orthography to phonology, Seidenberg & McClelland (1989) argue that, under some circumstances, normal subjects can perform lexical decision solely on the basis of orthographic or phonological "familiarity." In their model, orthographic familiarity is defined as the degree to which a letter string (word or non-word) can be re-created from the internal representation it generates. Phonological familiarity as a basis for lexical decision is more problematic as it depends on the ability of the network to generate the correct pronunciations of both words and non-words, which at least for non-words is less than satisfactory (Besner et al., 1990). Nonetheless, Seidenberg & McClelland demonstrate that their *undamaged* model is capable of distinguishing most words from orthographically regular non-words on the basis of orthographic familiarity.

These results suggest that some measure of orthographic familiarity in the DBM network might provide a basis for lexical decision. The DBM network was given connections among grapheme units and trained to generate orthography from semantics so that it would learn the orthographic structure among words in the same way as for semantic and phonological structure. However, if the network is to be required to actually *re-create* orthography, we cannot present input by clamping the grapheme units into their correct states as in previous simulations.¹³ Rather, we must provide the grapheme units with external input and require them to update their states in the same way as other units in the network. This is the same "soft clamping" technique that was used to train the phonological clean-up pathways of the IP and noIP output networks (see Section 3.2.2). Specifically, we presented a letter string to the network by providing each grapheme unit with fixed external input sufficient to generate a state of 0.9 if its desired state was 1, or -0.9 if its desired state was -1 . The initial states of grapheme units were set to 0.0 and updated over iterations just like the rest of the units in the network. The external input to grapheme units does not uniquely determine their final states because they also receive input from each other and from semantics via the intermediate units throughout the course of settling.

¹³Seidenberg & McClelland avoid this issue by training their network to re-generate orthography over a *separate* group of orthographic units from the ones used to present input.

We used as a measure of familiarity of a letter string the proximity between the desired states of the grapheme units and their final states after settling when presented with the letter string as external input. We will refer to this measure as “orthographic/semantic familiarity” because it reflects the consistency of a letter string with both of these types of knowledge. The undamaged network produces an orthographic/semantic familiarity greater than 0.995 (maximum 1.0) for 35 of the words—it fails on CAN, MAT, DOG, HAM and HOCK. These “misses” reflect the fact that the network was not trained with soft clamping. In contrast, only three of the non-words, all in the “close” set, are considered this familiar: DONE, MED and PIP. This performance yields a $d' = 2.59$ if this measure and criterion were adopted in a lexical decision task.

After damage to the network, the support that words receive from semantics is somewhat degraded and so we would expect the differences between words and non-words to be reduced. However, the network remains able to distinguish words from non-words fairly reliably. Considering all lesions producing correct performance between 20–80%, an orthographic/semantic familiarity criterion of 0.995 yields a $d' = 1.94$ overall ($d' = 1.51$ words vs. close non-words, 3.11 words vs. distant non-words). Reasonable lexical decision performance is retained even when explicit correct reading performance is below 40% ($d' = 1.67$ overall, 1.22 vs. close, 3.15 vs. distant) or when only word presentations resulting in omissions are included ($d' = 1.94$ overall, 1.51 vs. close, 3.12 vs. distant). Thus, similar to most deep dyslexics, the damaged network is able to distinguish words from non-words even under conditions when it cannot explicitly generate the pronunciation of the words.

5.5 Summary

The lesion experiments in this chapter attempt to serve three major purposes. The first is to demonstrate the generality of the H&S results across networks developed with very different learning procedures. The second is to support the use of back-propagation in cognitive modeling against criticisms based on its biological implausibility by providing evidence that the representations it develops have qualitatively similar properties to those developed with more plausible learning frameworks. The third is to illustrate how certain additional aspects of these alternative frameworks are particularly useful in understanding a number of characteristics of deep dyslexics.

The primary focus of the simulations presented in the paper thus far has been on demonstrating and understanding the degree to which the replication of deep dyslexic reading behavior in lesioned attractor networks depends on various aspects of their design. However, in many ways the empirical limitations of the original H&S model are more severe than its computational ones. Only the most basic aspects of the syndrome were modeled: the co-occurrence of semantic, visual, and mixed visual-and-semantic errors. Our simulations have extended the range of empirical phenomena that have been addressed to include additional error types, confidence ratings, and lexical decision.

However, there are fundamental characteristics of the patients' reading behavior, such as effects of word imageability/concreteness and part-of-speech, that remain unaccounted for. These aspects of deep dyslexia simply could not be addressed using the H&S word set, which only contains concrete nouns. The next chapter presents simulations that attempt to overcome these limitations and extend the empirical adequacy of attractor networks for modeling deep dyslexia.