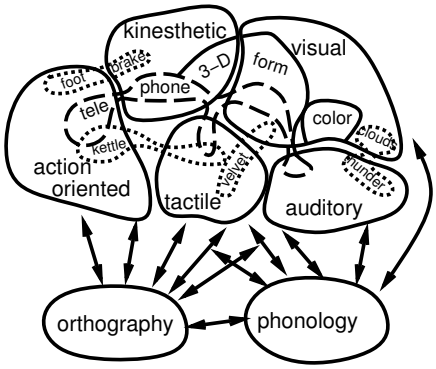

Figure 10.23: Illustration of how semantic information might be distributed across a number of more specific processing pathways, represented here by different sensory modalities. Linguistic representations (orthography, phonology) are associated with corresponding distributed activation patterns across semantics. Figure adapted from Allport, 1985.

using the mechanistic principles in Leabra that are well motivated for a number of other reasons. The complexity of the model and the task(s) it performs may need to be increased to better fit the human data.

10.6 Semantic Representations from Word Co-occurrences and Hebbian Learning

Our next model explores the development of the semantic representation component of the distributed lexicon framework based on statistical properties of the linguistic inputs. As discussed in chapter 7, semantic representations, like word representations, are viewed as distributed throughout various specialized brain areas. This notion of distributed semantics is well captured by figure 10.23, adapted from Allport (1985), which shows the representations of various items (a telephone, velvet, etc.) across several different specialized processing pathways. Although these pathways are represented only by sensory modality in the figure, a number of other types of specializations (e.g., chronological, sequential, task-oriented, episodic, etc.) probably also contribute to the overall distributed representation of an item. Indeed, essentially any pattern of activity anywhere in the brain that contributes some item-specific information could be considered a semantic representation.

One influential neural network model of human semantic representations (Farah & McClelland, 1991) used basic neural network principles to explain a somewhat puzzling pattern of dissociations across patients with different kinds of semantic deficits. The dissociations involve selective deficits for knowledge about living things versus nonliving things. Different groups of patients have deficits in one of these categories, but are relatively spared in the other (i.e., a *double-dissociation*; Warrington & Shallice, 1984; Warrington & McCarthy, 1983, 1987). One natural conclusion from this data is that human semantic information is organized according to the kinds of taxonomic categories that one might use in playing 20 questions: e.g., animal, vegetable, or mineral/man-made (Hillis & Caramazza, 1991).

Alternatively, as first proposed by Warrington and colleagues and subsequently elaborated by Farah and McClelland (1991), category-specific semantic deficits might arise based on the *sensory* versus *functional* associations with living versus nonliving things. People predominantly list sensory features when describing living things (e.g., "brown," "large" for a bear), and functional features when describing tools and other artifacts (e.g., "used for pounding in nails" for a hammer) (Farah & McClelland, 1991). Thus, the living-thing deficit could be explained by damage to sensory semantic areas, and the nonliving-thing deficit could be explained by damage to functional semantic areas. This account is much more consistent with the kind of distributed semantics envisioned in figure 10.23, which builds off of the well-known specialization of brain areas for different sensory modalities.

The simulation of this sensory–functional idea not only captured the basic dissociation between living and nonliving things, it also resolved an apparent difficulty for this approach quite naturally. Specifically, the verbal (non-implemented) sensory–functional account appeared to mistakenly predict that patients with the living-thing deficit should still show intact functional semantics for living things (whereas patients with the nonliving-thing deficit should still show intact sensory semantics for nonliving things). This pattern does not

appear to hold in the patients, however. The model implementing the sensory–functional idea demonstrated that the kinds of mutual support dynamics that emerge from bidirectionally connected neural networks (chapter 3) can account for this effect: after the functional semantics associated with living things lost their mutual support from the previously stronger sensory semantics representations, they became much more difficult to activate and therefore showed an impairment, even though the functional representations themselves were intact. In summary, the Farah and McClelland (1991) model showed how distributed semantic representations can exhibit counterintuitive dynamics under damage, in a manner consistent with the observed neuropsychological data.

The Farah and McClelland (1991) model is consistent with all the basic principles of our framework, but it does not address the question of how semantic representations develop from experience in the first place, and it cannot be easily probed using our own commonsense intuitions about semantics because it used random semantic patterns (much like we did above in the past-tense model). Therefore, we instead explore a model that implements just one piece of the larger distributed semantic network but allows us to see how semantic representations can develop in the first place, and to explore the properties of these representations using our own familiar intuitions about semantic meaning.

The semantic representations in our model emerge from simply accumulating information about which words tend to occur together in speech or reading. This model is based on the ideas of Landauer and Dumais (1997), who have developed a method they call *latent semantic analysis* or *LSA* that is based on computing the principal components (using a variant of PCA; see chapter 4) of word co-occurrence statistics. They have shown that just this co-occurrence information can yield semantic representations that do a surprisingly good job at mimicking human semantic judgments.

For example, an LSA system trained on the text of a psychology textbook was able to get a passing grade on a multiple choice exam based on the text. Although the performance (roughly 65–70% correct) was well short of that of a good student, it is nonetheless surprising that a simple automated procedure can perform as well as it does. LSA has also been used to perform automated essay grading, by comparing the semantic representation of a student's essay with those of various reference essays that have received different human grades. The correlation between grades assigned by LSA and those of human graders was the same as that between different human graders.

The word co-occurrence approach captures word *association* semantics, not just *definitional* semantics. For example, the definition of the word "bread" has nothing to do with the word "butter," but these two words are highly linked associationally. It appears that these associational links are important for capturing the structure of human semantic memory. For example, in semantic priming studies, the word "butter" is read faster when preceded by the word "bread" than when preceded by an unrelated word (e.g., "locomotive").

As you might expect from chapter 4, Hebbian model learning provides a natural mechanism for learning word co-occurrence statistics. Indeed, CPCA Hebbian learning is closely related to the the sequential principal components analysis (SPCA) technique that the LSA method is based on. Interestingly, the network and training we use here are essentially identical to those used in the Hebbian model of receptive field development in the early visual system, as described in chapter 8. In both of these models, the network extracts the reliable correlations from naturalistic input stimuli using a simple Hebbian learning mechanism.

This type of learning requires a sufficiently large sample of text for the extraction of useful statistics about which words tend to co-occur. A particularly convenient source of such text is this textbook itself! So, we trained a simple CPCA Hebbian network by having it "read" an earlier draft of this textbook paragraph-by-paragraph, causing it to represent the systematic patterns of co-occurrence among the words. In the following exploration, we then probe the resulting network to see if it has captured important aspects of the information presented in this text.

One of the most important themes that emerges from this exploration, which is highly relevant for the larger distributed semantics framework, is the power of distributed representations to capture the complexity of semantic information. We will see that the distributed and

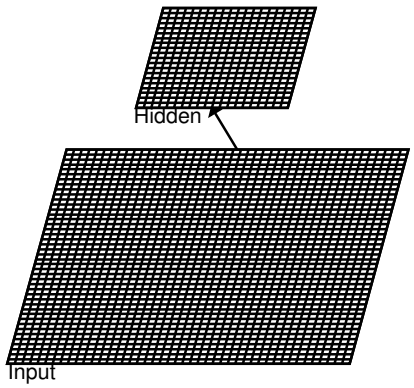

Figure 10.24: Semantic network, with the input having one unit per word. For training, all the words in a paragraph of text are activated, and CPCA Hebbian learning takes place on the resulting hidden layer activations.

competitive activation dynamics in the network can perform multiple constraint satisfaction to settle on a pattern of activity that captures the meaning of arbitrary groups of words, including interactions between words that give rise to different shades of meaning. Thus, the combinatorics of the distributed representation, with the appropriate dynamics, allow a huge and complex space of semantic information to be efficiently represented.

10.6.1 Basic Properties of the Model

The model has an input layer with one unit per word (1920 words total), that projects to a hidden layer containing 400 units (figure 10.24). On a given training trial, all of the units representing the words present in one individual paragraph are activated, and the network settles on a hidden activation pattern as a function of this input. If two words reliably appear together within a paragraph, they will be semantically linked by the network. It is also possible to consider smaller grouping units (e.g., sentences) and larger ones (passages), but paragraph-level grouping has proven effective in the LSA models. After the network settles on one paragraph's worth of words, CPCA Hebbian learning takes place, encoding the conditional probability that the input units are active given that the hidden units are. This process is repeated by cycling through all the paragraphs in the text multiple times.

The development of effective semantic representations in the network depends on the fact that whenever two similar input patterns are presented, similar hidden units will likely be activated. These hidden layer representations should thus encode any reliable correlations across subsets of words. Furthermore, the network will even learn about two words that have similar meanings, but that do not happen to appear together in the same paragraph. Because these words are likely to co-occur with similar other words, both will activate a common subset of hidden units. These hidden units will learn a similar (high) conditional probability for both words. Thus, when each of these words is presented alone to the network, they will produce similar hidden activation patterns, indicating that they are semantically related.

To keep the network a manageable size, we filtered the text before presenting it to the network, thereby reducing the total number of words represented in the input layer. Two different types of filtering were performed, eliminating the least informative words: high-frequency noncontent words and low frequency words. The high-frequency noncontent words primarily have a syntactic role, such as determiners (e.g., "the," "a," "an"), conjunctions ("and," "or," etc.), pronouns and their possessives ("it," "his," etc.), prepositions ("above," "about," etc.), and qualifiers ("may," "can," etc.). We also filtered transition words like "however" and "because," high-frequency verbs ("is," "have," "do," "make," "use," and "give"), and number words ("one," "two," etc.). The full list of filtered words can be found in the file `sem_filter.lg.list` (the "lg" designates that this is a relatively large list of high-frequency words to filter) in the `chapter_10` directory where the simulations are.

The low-frequency filtering removed any word that occurred five or fewer times in the entire text. Repetitions of a word within a single paragraph contributed only once to this frequency measure, because only the single word unit can be active in the network even if it occurs multiple times within a paragraph. The frequencies of the words in this text are listed in the file `eccn_lg_f5.freq`, and a summary count of the number of words at the lower frequencies (1–10) is given in the file `eccn_lg_f5.freq_cnt`. The ac-

tual filtered text itself is in `eccn_lg_f5.cln`, providing a sense of the "telegraphic" nature of the input the network receives.

In addition, the network has no sense of word order or any other syntactic information, so it is really operating at the level of the "gist." That the network is operating on such minimal input makes its performance all the more impressive. Nevertheless, achieving substantially better levels of comprehension will likely require a substantially more complex network that processes more of the information available in real texts.

10.6.2 Exploring the Model

[Note: this simulation requires a minimum of 128Mb of RAM to run.]

↪ Open the project `sem.proj.gz` in `chapter_10` to begin.

As before, the network is a skeleton, and must be built and trained weights loaded.

↪ Do `LoadNet` on the `sem_ctrl` control panel.

Individual Unit Representations

To start, let's examine the weights of individual units in the network.

↪ Select `r.wt`, and then select various hidden units at random to view.

You should observe sparse patterns of weights, with different units picking up on different patterns of words in the input. However, because the input units are too small to be labeled, you can't really tell which words a given unit is activated by. The `GetWordRF` button (get receptive field in terms of words) on the `sem_ctrl` control panel provides a work-around to this problem.

↪ View the weights for the lower-leftmost hidden unit, and then hit the `GetWordRF` button.

A `String_Array` window will pop up, containing an alphabetized list of the words that this hidden unit receives from with a weight above the `rf_thresh` of .5 (table 10.11). You can resize this window to see more of the words at one time, and you can use the middle mouse button to scroll the text within one field.

One of the most interesting things to notice here is that the unit represents multiple roughly synonymous terms. For example, you should see the words "act," "activation," and "activations," in the fields numbered 0–2 in the window. By scrolling through this list you should see many more examples of this.

act activation activations algorithm allow already approaches arbitrary areas argue artificial aspect assess atoms balance based basic behavior bias biological biologically biology bit body calcium called capable categories cell channel channels charge chl closed cognition combination combine complicated components computational compute computed concentrations conductance confused consider consistent constantly contain continuous correlations corresponding cpca critical cross current currents demands derivative described detailed detector detectors determined difference different diffusion discussed divide division electrical electricity emphasize encoded enter environment equal equation etc exceeds excitatory explain extent family fast favor fires firing flow follow forces form function functioning generally generec hand hebbian help hypothesis identify imagine implement implementing important include including inconsistent indicated individual influence information inhibitory input inputs integrates integrating integration interactivity interested ion ions kind labor language largely last later leak learning leaving let level likelihood liquid magnitude major manner matches mathematically matter mechanisms membrane memories minus model modeling models modification movement name need negative net network networks neural neuron neurons non notice now number numbers occur open opened opposite oriented parallel pass pattern phase phonology picture plus positive possibly potassium potential practical prevents principles processing properties purposes put rapid rate reasons recall referred reflected reflects relevant remains research responding result rule same saw say see sends separate showed shown sign simple slow soft special specific specifically states strongest suggested summarized summary survival synaptic textbook things think threshold time times towards tradeoff turn type types typically understanding updated variable versa version via vice voltage vowel weights work world writing

Table 10.11: List of words with weights $> .5$ for the lower-leftmost hidden unit, as produced by the `GetWordRF` button.

Question 10.10 *List some other examples of roughly synonymous terms represented by this unit.*

This property of the representation is interesting for two reasons. First, it indicates that the representations are doing something sensible, in that semantically related words are represented by the same unit. Second, these synonyms probably do not occur together in the same paragraph very often. Typically, only one version of a given word is used in a given context. For example, "The activity of the unit is..." may appear in one paragraph, while "The unit's activation was..." may appear in another. Thus, for such representations to develop, it must be based on the similarity in the general contexts in which similar words appear (e.g., the co-occurrence of "activity" and "activation" with "unit" in the previous example). This generalization of the semantic similarity structure across paragraphs is essential to enable the

network to transcend rote memorization of the text itself, and produce representations that will be effective for processing novel text items.

↪ View the GetWordRF representations for several other units to get a sense of the general patterns across units.

Although there clearly is sensible semantic structure at a local level within the individual-unit representations, it should also be clear that there is no single, coherent theme relating all of the words represented by a given unit. Thus, individual units participate in representing many different clusters of semantic structure, and it is only in the aggregate patterns of activity across many units that more coherent representations emerge. This network thus provides an excellent example of distributed representation.

Distributed Representations via Sending Weights

A more appropriate technique for exploring the nature of the distributed semantic representations is to look at the sending weights from a given word to the hidden layer. This will show the pattern of hidden units that represent that word, and by doing this for multiple different words, we can get a sense of how many hidden units the words have in common. In other words, we can look at distributed pattern overlap as a measure of the similarity structure (as discussed in chapter 3).

↪ Press SelWord on the sem_ctrl control panel. Enter "attention" as the word, and click both the view and selwts buttons on.

This will allow us to view the weights for the input unit corresponding to the word "attention," and it will select all of the hidden units which have a weight value greater than the rf_thresh of .5.

↪ You have to select s.wt in the network to actually view the sending weights.

You should see a pattern of hidden units that have strong weights from this word unit, with the strongest units selected (indicated by the white dashed line surrounding the unit). By itself, this pattern is not very meaningful. However, we can now compare the pattern with that of another, related word.

↪ Do SelWord again, but enter in "binding" as the word this time, and click the selwts button off.

Event	attenti	binding	dyslexi
attention	1	0.415	0.090
binding	0.415	1	0.118
dyslexia	0.090	0.118	1

Table 10.12: Cosine matrix of the hidden representations for the words "attention," "binding," and "dyslexia".

By turning selwts off, we ensure that the previously selected units from "attention" will remain so. Now, you should be able to see that in several cases, the strong weights for "binding" overlap with those of the selected units from "attention." Thus, these overlapping units represent both of these concepts, as one might expect given that they have a clear semantic relationship. Now let's see what a more unrelated word looks like.

↪ Do SelWord again, and enter "dyslexia" as the word, and keep the other buttons as they are.

You should now observe considerably less overlap among the strong weights from "dyslexia" and the selected units from "attention," which is appropriate given that these words are not as closely related.

The SelWord process can be performed automatically for two different words by using the PairProbe button. The first word's weights will be selected, and the second word's weights will be viewed as color values.

↪ Do some PairProbes for other word pairs that you expect to be semantically related in this textbook, and pairs that you expect to be unrelated.

Summarizing Similarity with Cosines

Instead of just eyeballing the pattern overlap, we can compute a numerical measure of similarity using *normalized inner products* or *cosines* between pairs of sending weight patterns (see equation 10.1 from the dyslexia model in section 10.3). The WordMatrix button does this computation automatically for a list of words (separated by spaces), producing a matrix of all pairwise cosines, and a cluster plot of this distance matrix.

↪ Do WordMatrix, and enter "attention binding dyslexia" as the words.

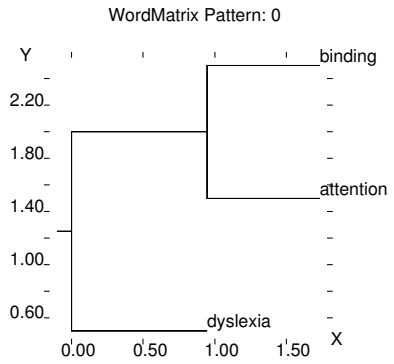

Figure 10.25: Cluster plot of the similarity structure for attention, binding, and dyslexia as produced by the `WordMatrix` function.

You should replicate the same basic effect we saw above — attention and binding are more closely related to each other than they are to dyslexia. This can be seen in the cluster plot (figure 10.25) by noting that attention and binding are clustered together. The cosine matrix appears in the terminal window where you started the program (it should look like table 10.12). Here, you can see that the cosine between "attention" and "binding" is .415, (relatively high), while that of "attention" and "dyslexia" is only .090, and that between "binding" and "dyslexia" is only .118.

↪ Do a `WordMatrix` for several other words that the network should know about from "reading" this textbook.

Question 10.11 (a) *Report the cluster plot and cosine matrix results.* **(b)** *Comment on how well this matches your intuitive semantics from having read this textbook yourself.*

Distributed Representations via Activity Patterns

To this point we have only used the patterns of weights to the hidden units to determine how similar the semantic representations of different words are. We can also use the actual pattern of activation produced over the hidden layer as a measure of semantic similarity. This is important because it allows us to present multiple word inputs at the same time, and have the network choose a hidden layer representation that best fits this combination of words. Thus, novel semantic representations can be produced as combinations of semantic representations for individual words. This ability is critical for some of the more interesting and powerful applications of these semantic representations (e.g., multiple choice question answering, essay grading, etc.).

The `ActProbe` function can be used to do this activation-based probing of the semantic representations.

↪ Select `act` in the network window. Press `ActProbe` on the `sem_ctrl` control panel, and you will be prompted for two sets of words. Let's start with the same example we have used before, entering "attention" for the first word set, and "binding" for the second.

You should see the network activations updating as the word inputs are presented. The result pops up in a window, showing the cosine between the hidden activation patterns for the two sets of words. Notice that this cosine is lower than that produced by the weight-based analysis of the `WordMatrix` function. This can happen due to the activation dynamics, which can either magnify or minimize the differences present in the weights.

Next, let's use `ActProbe` to see how we can sway an otherwise somewhat ambiguous term to be interpreted in a particular way. For example, the term "attention" can be used in two somewhat different contexts. One context concerns the implementational aspects of attention, most closely associated with "competition." Another context concerns the use of attention to solve the binding problem, that is associated with "invariant object recognition." Let's begin this exploration by first establishing the baseline association between "attention" and "invariant object recognition."

↪ Do an `ActProbe` with "attention" as the first word set, and "invariant object recognition" as the second.

You should get a cosine of around .302. Now, let's see if adding "binding" in addition to "attention" increases the hidden layer similarity.

↪ Do an `ActProbe` with "attention binding" as the first word set, and "invariant object recognition" again as the second.

The similarity does indeed increase, producing a cosine of around .326. To make sure that there is an in-

teraction between "attention" and "binding" producing this increase, we need to test with "binding" alone.

↪ Do an `ActProbe` with "binding" as the first word set, and "invariant object recognition" again as the second.

The similarity drops back to .288. Thus, there is something special about the combination of "attention" and "binding" together that is not present by using each of them alone. Now if we instead probe with "attention competition" as compared to "invariant object recognition," we should activate a different sense of attention, and get a smaller cosine.

↪ Do an `ActProbe` with "attention competition" as the first word set, and "invariant object recognition" again as the second.

The similarity does now decrease, with a cosine of only around .114. Thus, we can see that the network's activation dynamics can be influenced to emphasize different senses of a word. Thus, this is potentially a very powerful and flexible form of semantic representation that combines rich, overlapping distributed representations and activation dynamics that can magnify or diminish the similarities of different word combinations.

Question 10.12 *Think of another example of a word that has different senses (that is well represented in this textbook), and perform an experiment similar to the one we just performed to manipulate these different senses. Document and discuss your results.*

A Multiple-Choice Quiz

Finally, we can run an automated multiple-choice quiz on the network. We created ten multiple-choice questions, shown in table 10.13. Note the telegraphic form of the quiz, as it contains only the content words that the network was actually trained on. The best answer is always *A*, and *B* was designed to be a plausible foil, while *C* is obviously unrelated (unlike people, the network can't pick up on these regularities across test items). The quiz is presented to the network by first presenting the "question," recording the resulting hidden activation pattern, and then presenting each possible answer and computing the cosine of the resulting hidden activation with that of the question. The answer that has the closest cosine is chosen as the network's answer.

0.	neural activation function
A	spiking rate code membrane potential point
B	interactive bidirectional feedforward
C	language generalization nonwords
1.	transformation
A	emphasizing distinctions collapsing differences
B	error driven hebbian task model based
C	spiking rate code membrane potential point
2.	bidirectional connectivity
A	amplification pattern completion
B	competition inhibition selection binding
C	language generalization nonwords
3.	cortex learning
A	error driven task based hebbian model
B	error driven task based
C	gradual feature conjunction spatial invariance
4.	object recognition
A	gradual feature conjunction spatial invariance
B	error driven task based hebbian model
C	amplification pattern completion
5.	attention
A	competition inhibition selection binding
B	gradual feature conjunction spatial invariance
C	spiking rate code membrane potential point
6.	weight based priming
A	long term changes learning
B	active maintenance short term residual
C	fast arbitrary details conjunctive
7.	hippocampus learning
A	fast arbitrary details conjunctive
B	slow integration general structure
C	error driven hebbian task model based
8.	dyslexia
A	surface deep phonological reading problem damage
B	speech output hearing language nonwords
C	competition inhibition selection binding
9.	past tense
A	overregularization shaped curve
B	speech output hearing language nonwords
C	fast arbitrary details conjunctive

Table 10.13: Multiple-choice quiz given to the network based on this text. The network compares the activation pattern for each answer with that of the question and selects the closest fitting one.

↪ To run the quiz, first open up a log by doing `View`, `QUIZ_LOG`. Then, open up a process control panel by doing `View`, `QUIZ_PROCESS_CTRL`. The `NEpoch_2` process control panel will appear. Do a `ReInit` and a `Step`.

This presents the first question to the network ("neural activation function").

↪ `Step` 3 more times for each of the possible answers.

After the last step, you should see the `Epoch_2_TextLog` update, with a record of the cosine distances for each of the answers compared to the question, and the answer that the network came

up with (*B* in this case, though the correct answer *A* was considerably closer than the clearly wrong answer, *C*). Now let's just run through the rest of the quiz.

↪ Turn off the network `Display` toggle (upper left-hand corner of the NetView), and `Run` the `NEpoch_2` process through the remainder of the quiz.

You should observe that the network does OK, but not exceptionally — 60–80 percent performance is typical. Usually, the network does a pretty good job of rejecting the obviously unrelated answer *C*, but it does not always match our sense of *A* being better than *B*. In question 6, the *B* phrase was often mentioned in the context of the question phrase, but as a *contrast* to it, not a similarity. Because the network does not have the syntactic knowledge to pick up on this kind distinction, it considers them to be closely related because they appear together. This probably reflects at least some of what goes on in humans — we have a strong association between "black" and "white" even though they are opposites. However, we can also use syntactic information to further refine our semantic representations — a skill that is lacking in this network. The next section describes a model that begins to address this skill.

↪ Go to the `PDP++Root` window. To continue on to the next simulation, close this project first by selecting `.projects/Remove/Project_0`. Or, if you wish to stop now, quit by selecting `Object/Quit`.

10.6.3 Summary and Discussion

Perhaps the most important aspects of this semantics model are its ability to represent variations in semantic meaning in terms of rich, overlapping distributed representations, and its ability to develop these representations using Hebbian learning. Because our low-level visual representations model from chapter 8 relies on these same features, this helps to reaffirm our belief that a common set of principles can be used to understand cortical processing across a wide range of different domains, from visual perception to language in this case.

One limitation of this model is that it does not include any task-based learning. The inclusion of this type of learning might help to further refine the distinctions captured by the semantic representations, and improve performance on things like the multiple choice problems. Indeed, the addition of such task-based learning might serve as a useful model of the beneficial effects of production on comprehension. We have probably all had the experience of feeling as if we understood something, only to find that this feeling was somewhat illusory when it came time to describe the idea orally or in writing. Similarly, the present model has a fairly fuzzy set of representations based on its purely passive, receptive experience with the domain. If we included task-based learning as well, this would be like giving the model practice (and feedback) articulating various ideas, which would presumably clarify and sharpen the representations.

Although the model we explored demonstrates a number of important principles, it falls short of addressing the issue of the topographic organization of different kinds of semantic information across different brain areas. As mentioned in the introduction, this organizational issue has been a major focus of neuropsychological studies, and neural network models such as the one by Farah and McClelland (1991) have played an important, if controversial, role in this field. One could consider extending our model by including recurrent self-connections within the hidden layer, and exploring the topographic organization that develops as a result, much in the same way we explored the topographic organization of visual feature detectors in chapter 8.

However, we are somewhat skeptical of such an approach, because it seems clear that the topographic organization in the human brain depends on all kinds of constraints that might be difficult to capture in such a simple model. Referring back to figure 10.23 and the distributed semantics idea of Allport (1985), it would seem that one would have to build models that include sufficient instantiations of various sensory systems, together with language processing and functional motor-based processing. While not impossible, this is a more formidable challenge that awaits future ambitions.

10.7 Sentence-Level Processing

The semantic representations model we just explored provides one step toward understanding language processing at a level higher than individual words. However, the semantic model was limited in that it com-