The vector-coding account of human facial recognition, appealing as it is, must surmount a serious objection. Humans do not possess twenty or so distinct types of sensory cells, each sensitive to one specific aspect of any perceived face. We have no sensory analog to the four types of taste receptors or the six types of smell receptors. We have only eyes to code faces at our sensory periphery, and the cells on the retina are sensitive to color and brightness and changes therein, but they couldn't care less about faces. How then do we manage to represent and recognize faces?

Where and How Are Faces Coded?

The account to be outlined below is still conjectural rather than proven, but it is a plausible account of how humans recognize faces and, equally important for our purposes here, it is a highly *accessible* example of how vector processing works. The first part of the idea is this. Although no retinal cell is responsive specifically to any of the various aspects of a face that are relevant for facial recognition, collectively the retinal cells do contain information about perceived faces as an implicit part of their overall pattern of cellular activations. Moreover, they do send that implicit information forward to subsequent populations of neurons—to the LGN cells, to the visual cortex, and ultimately to a special area in the temporal lobe that is crucial for facial recognition. Is it possible that the cell types that code faces explicitly are to be found not at the body's sensory periphery, as with taste and smell, but farther along in the chain of cell populations?

Not only is it possible; apparently it is actual. Isolated physical damage to a specific area of the brain's temporal lobe—the result of a tumor, perhaps, or a stroke (a burst blood vessel)—produces a strange condition known as *facial agnosia*. Familiar to neurologists working the hospital wards, the occasional patients with this odd affliction show a highly specific loss of the normal ability to recognize *faces*, even those previously well known to them. Nor can

the patient learn to recognize any new faces. Surprisingly, there is nothing wrong with the patient's eyes, and he can visually recognize most nonfacial objects without any pause or difficulty. But the face of his brother, or his wife, or even his own face in a mirror, evokes no recognition. He may easily identify these people by their voices, by their dress, or by some other cue. But the distinctive character of their faces, and of all other faces, is now and forever beyond his visual grasp.

So there does appear to be a distinct population of neurons specialized for the coding of faces, a population perhaps five or six synaptic steps downstream from the retina.

The second part of the idea is this. The many synaptic connections, between the retinal cells and the distant "facial cells" in the temporal lobe, filter and transform the incoming information in such a way that the "facial" cells respond to, and only to, the important dimensions of facial structure implicitly coded in the overall retinal-activation pattern. The retinal cells collectively contain oceans of information, of course: about trees and benches and stoplights and doors. But the special connective path, leading stepwise from the eye to the facial area described above, suppresses or ignores all of that information, except for such *facial* features as may happen to be retinally represented along with everything else. To exactly these features, diffuse and implicit though their retinal representation may be, the downstream population of neurons responds vigorously.

Simple Pattern Recognition

How is such selective magic possible? A fully general answer is impossible to state in a few sentences, but a first approximation to the correct answer is easily given. Indeed, we can display it visually. Let us take, for our example, something a little simpler than a face. Suppose we wish to discriminate the occasional registration of the letter "T" on a small screen of exactly nine light-sensitive cells or pixels. (See figure 3.1. For ease of visual apprehension, we shall suppose that the blackened areas are the ones receiving illumination.)

We can achieve this goal by funneling the nine output axons of those small retinal cells to a single large target cell, there to make nine synaptic connections all of the same size or "weight," but differing in their several polarities. The job is completed by making

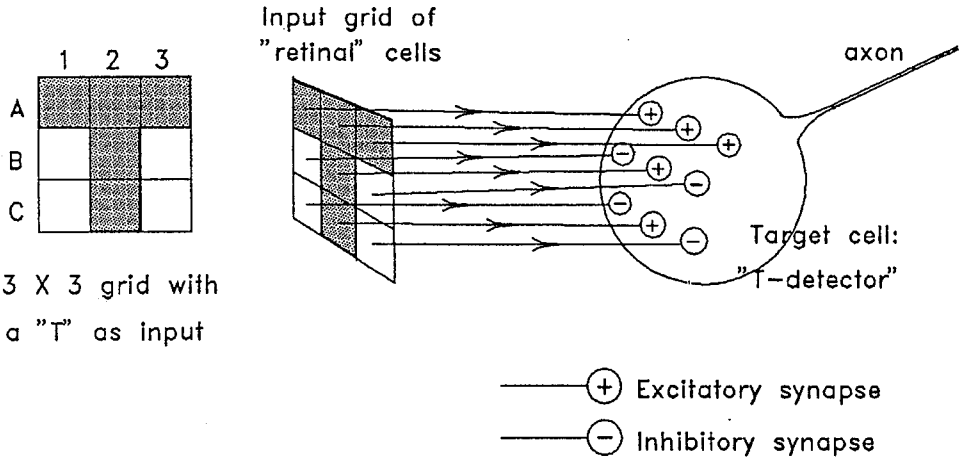

Figure 3.1 A simple arrangement for pattern recognition. The cell at the right is maximally activated when, and only when, a T is projected onto the grid of "retinal" cells at the left.

the connections from cells A1, A2, A3, B2, and C2 all positive or excitatory, and by making the connections from cells B1, C1, B3, and C3 all negative or inhibitory. The only possible way to activate the target cell, then, is to activate one or more of the first group of cells: the T-element cells. As each one of those five cells becomes activated by illumination, the excitatory effect on the target cell goes up. Maximum activation is achieved when all of the T-element cells are illuminated, *unless* one or more of the non-T cells is also illuminated. That would have the effect of inhibiting the target cell to the appropriate degree, and of reducing its level of activation.

What we have here is a simple tug-of-war between the excitatory effects of the T-element cells and the inhibitory effects of the non-T cells. So long as there are conflicting messages reaching the target cell, its activation level will be below its maximum level. The inhibitory messages will always cancel out a corresponding number of excitatory messages, if any there should happen to be. Only when the entire retinal grid is unanimous in its collective message—the T-element cells are all on, and the non-T cells are all off—will the target cell reach its maximum level of activation.

The target cell has thus become a T-detecting cell. Although none of the individual retinal cells knew or cared about the projection of a T across their population, the target cell *does* care. It cares most about perfect Ts, complete in every respect and unmarred by additional illuminations elsewhere in its "visual field." There is a technical term for the input pattern that a cell likes best: that pattern is

called the *preferred stimulus* of the cell in question. This term allows us a convenient way of talking about the discriminatory focus of an individual cell. We simply specify its preferred stimulus.

It is important to notice that our artfully tuned target cell will also respond fairly strongly to patterns of illumination that are very close to a perfect T: to a T with a single pixel missing, for example, or to an otherwise perfect T marred by the illumination of some pixel outside the perfect pattern. The target cell's response will be suboptimal, but it will still be considerable. Accordingly, our target cell will indicate not only the occurrence of the occasional bull's eye, it will give a graded indication of the occurrence of anything closely *similar* to a perfect T.

Notice further that we could just as easily have made the target cell a detector of U-shaped patterns, or of L-shaped patterns, or of O-shaped patterns, and so on. Simply arrange appropriately the pattern of excitatory and inhibitory synaptic connections projecting from the retinal grid to the target cell (see again figure 3.1), and you can have it detect any of the 2^9 distinct patterns possible for our nine-element retinal grid.

More important still, we can put in place an entire population of downstream target cells, each one of which receives a complete set of axonal end-branches from the projecting retinal axons, and each one of which has a configuration of excitatory and inhibitory synaptic connections that makes it a detector of exactly one of the possible patterns. This downstream population of target cells will thus constitute a well-informed committee, with each member responsible for the detection of some specific pattern. Complex patterns that were lost on the individual retinal cells will thus be registered with keen and focused concern by the cells at the downstream population. In this way can distributed features at the input level be unerringly "picked out" at subsequent levels of neurons, thanks to the filtering activity of an artfully arranged set of synaptic connections. This is an important reason—the first of many reasons—why coding vectors at the sensory periphery often need to be processed and transformed by passing them through a matrix of synaptic connections.

Face Recognition

We were talking about face recognition. Let us now return. The lessons just learned about pattern recognition apply equally well

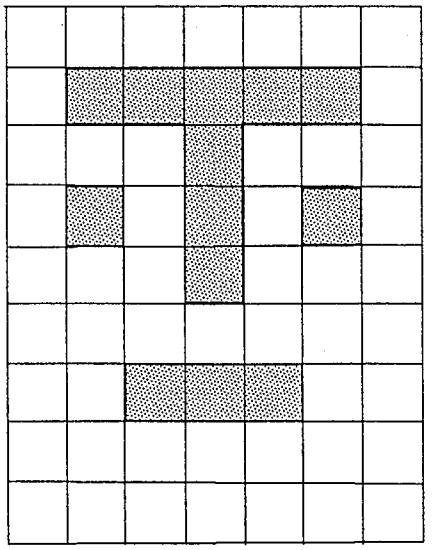

Figure 3.2 A rudimentary face pattern in a 9 × 7 grid.

here. See the slightly more complex pattern displayed in figure 3.2. Here again we have a T pattern, this time flanked by a pair of eyes and underlined by a mouth. The T thus appears as eyebrows bisected by a nose. This facelike pattern is as readily detectable by some downstream target cell as was the letter T in our earlier example. The retinal grid is now nine cells by seven, rather than three by three, but the principle of downstream detection is the same. We need axonal projections, from all sixty-three of these sensory cells, to an appropriate target cell. The retinal cells darkened in figure 3.2 will need excitatory projections to the target cell, and the undarkened cells will need inhibitory projections. A target cell thus connected will serve as a primitive *face* detector as neatly as you please. Its preferred stimulus—the specific input pattern that produces its maximum level of activation—is precisely the face of figure 3.2. Notice finally that this system will register the occurrence of that face, or of something similar to it, in no more time than it takes the information to traverse the axons and the synaptic connections involved: in this case, about 1/100th of a second.

Admittedly, this is still pretty thin fare. The face of figure 3.2 is rudimentary in the extreme, and the downstream target cell can indicate little more than its presence or absence. It cannot hope to recognize real faces, much less discriminate your brother from your sister. We need something more. Most obviously, we need to consider input grids with many thousands of pixel cells, and we need

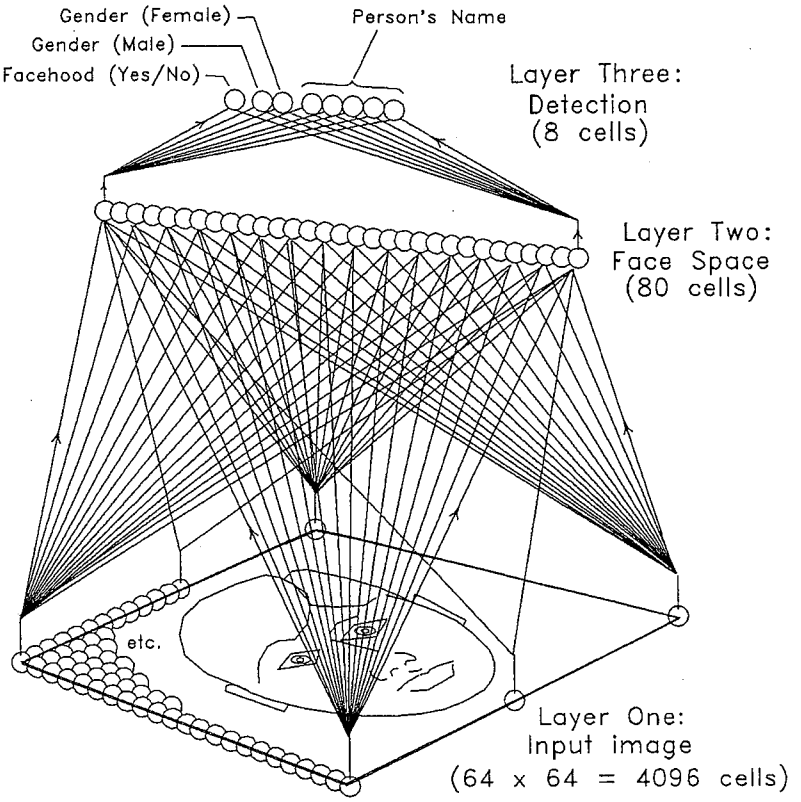

Figure 3.3 An artificial neural network for recognizing real faces.

pixels that will admit of a smooth variety of brightness levels rather than just "on" and "off." Less obviously, we need a largish *committee* of downstream cells that can detect a variety of distinct facial features within that larger retinal canvas. This would permit the coding of diverse faces within a set of proprietary dimensions; that is, within a "face space" of the general kind we explored in the previous chapter (figure 2.5). There we played with only three dimensions of variation across faces, and we could code only cartoons. A system adequate to represent and discriminate among real faces will need a good deal more.

This is precisely what some neural-net researchers have recently succeeded in modeling. (Thanks here to the work of Garrison Cottrell's group at the University of California, San Diego.) Their three-stage artificial network is schematically portrayed in figure 3.3. Its input layer or "retina" is a 64 × 64-pixel grid whose elements each admit of 256 different levels of activation or "brightness." This higher resolution, both in space and in brightness, allows the network to code recognizable representations of real faces. Figure 3.4 shows several of the many input photographs on which the net-

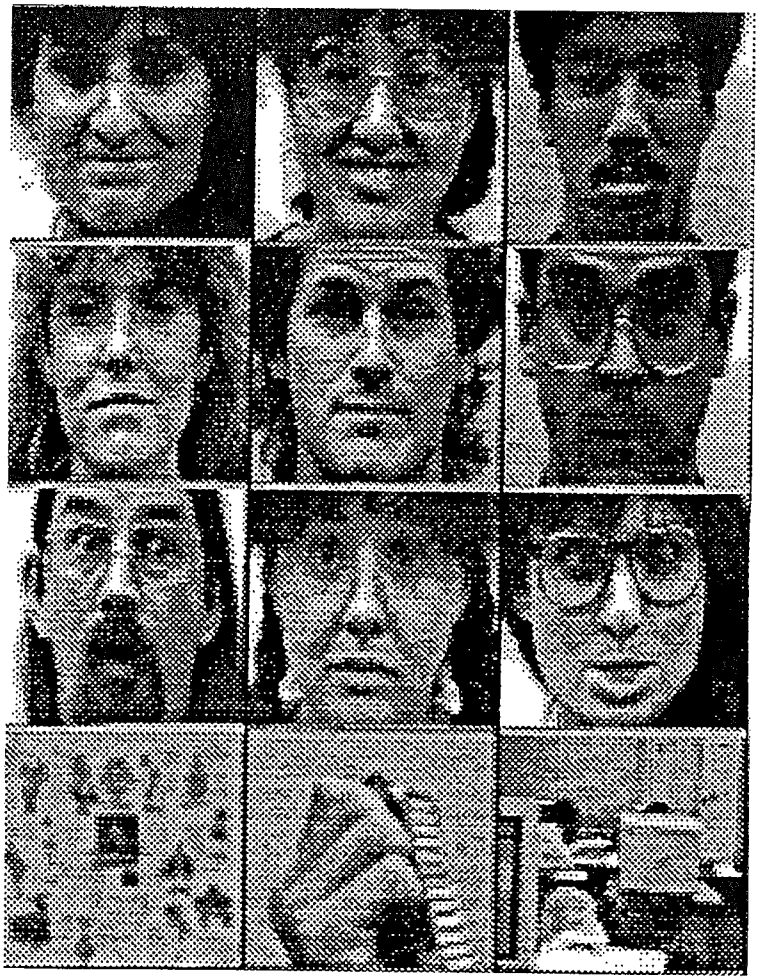

Figure 3.4 Selected input images for training the face-recognition network. (Thanks to Gary Cottrell.)

work was actually trained. That *training set* contained 64 different photos of 11 different faces, plus 13 photos of nonface scenes. Cottrell himself is the bespectacled face at the lower left.

Each input cell projects a radiating set of axonal end branches to each and every one of the 80 cells in the second layer, which layer represents an abstract space of 80 dimensions (rather than just three) in which the input faces are explicitly coded. This second layer projects finally to an output layer of only eight cells. These output cells have had their synaptic connections carefully tuned to discriminate, first, between faces and nonfaces; second, between male and female faces; and third, to respond with the person's "name" (an arbitrarily assigned numerical code) to the re-presentation of any face that the network "got to know" during its training period.

As with the simple T-pattern recognition system of figure 3.1, what actually does the work in Cottrell's face-recognition network is the overall configuration of synaptic connections—positive and negative, weak and strong. It is these, and only these, that progressively transform the initial 64 × 64-element pattern or vector into a second and finally a third vector that exactly represents the input photo's facehood, gender, and name explicitly. But while it was obvious how to configure the nine synaptic connections of the simple two-layer T-pattern recognizer, it is not at all obvious how to configure the connections in this much larger network. After all, it contains $(64 \times 64) + 80 + 8 = 4184$ cells, and a grand total of 328,320 synaptic connections!

Network Learning: Repeated Synaptic Adjustment

In fact, Cottrell's group had no idea how the synaptic connections should be configured. That was part of the problem they had to solve. Fortunately, there exists a general technique for finding synaptic solutions to transformational problems, a technique widely used by neuromodelers. It is biologically realistic in the single respect that it involves the steady adjustment of a network's synapses in response to the pressures of experience. Sadly, it is biologically unrealistic in just about every other respect. But beggars can't be choosers. Until we sleuth out the brain's actual learning procedures, we have to use whatever is available. This interim technique is called "synaptic adjustment by the successive backpropagation of errors," or just *backpropagation* for short. It has two outstanding virtues. It works like the dickens, at least for smallish networks. And the procedure can be administered by a conventional serial computer, which takes the extraordinary drudgery of the procedure out of the neuromodeler's hands. It works roughly as follows.

What does one wish one's network to do? To transform each one of a wide range of possible input vectors into an output vector that is somehow appropriate (depending on the particular skill being taught) for that input. And how does one do that? One sets the weights of the synaptic connections so that collectively they will perform the desired transformation. And if one does not know what weights to impose to achieve this end? One sets all of the synaptic weights at random values, both positive (excitatory) and negative

(inhibitory), not too far from zero on either side. This constitutes an utterly blind stab in the dark at the desired configuration of weights. By itself, this gets us nowhere, but have faith for a second.

What this stab permits is the discovery of just how *mistaken* it is, at least in the output vectors it is prone to produce. Let us therefore present one of the relevant input vectors to the input layer, and then observe the transformed result at the output layer. Given that the weights were set at random, almost certainly that particular output is gibberish, and not at all what we want the network to produce for that input. But at least we know what the appropriate output *should* have been. If the input face was Janet's, and Janet's name code is ".5, 1, .5, 0, 0," then the output vector ideally should have been {1; 0, 1; .5, 1, .5, 0, 0}. The first "1" codes that the input is indeed a face; the following "0, 1" codes "No" for male and "Yes" for female; and the rest identifies the face as Janet's.

Unfortunately, that is not the result we got. What we got was, say, {.23; .8, .39; .2, .03, .19, .66, .96}. But let us compare the desired to the actual output. Specifically, let us subtract each element of the actual vector from its corresponding element in the desired vector (figure 3.5). This gives us a third vector that represents exactly the *error* committed by the network for the input on this occasion. Now let us simply square each of these eight errormeasures. The point of doing this is to exaggerate somewhat the relative importance of the larger errors over the smaller ones. (The little ones we can tolerate; the big ones demand immediate attention.) The average of these eight squared errors is called the *mean squared error*. It is this number that we are determined to reduce.

And reduce it we can. We have to settle for a series of small, incremental reductions, but here is how it is done. Let us hold *all but one* of the network's synaptic weights *constant*, at their initial (randomly set) values, and then examine the contribution that this

	⟨ 1, 0, 1, .5, 1, .5, 0, 0 ⟩	Desired output vector
−	⟨ .23, .8, .39, .2, .03, .19, .66, .96 ⟩	Actual output vector
=	⟨ .77, −.8, .61, .3, .97, .31, −.66, −.96 ⟩	Error vector
	⟨ .59, .64, .37, .09, .94, .09, .44, .94 ⟩	Squared−error vector
	.5125 ←	Mean of squared errors

Figure 3.5 The calculation of the mean squared error at the output layer.

single isolated connection weight makes to the mean squared error. Since the network's transformational activities are entirely determined by its connective configuration, this smaller question is one we can answer decisively. In particular, we can determine whether a tiny change, up or down, in that isolated connection weight, would lead to an output vector *slightly* closer to the one desired. That is, we can determine whether a tiny change in that weight would reduce the mean squared error at the output. If it would make no detectable difference, we leave the weight as it is. If a tiny change upward (or downward) would improve things slightly, then we make that change, a change whose very small size is proportional to the degree of improvement promised.

The resulting change in the network's overall performance is of course minuscule. But having nudged one weight in a profitable or error-reducing direction, we shift our attention to the connection weight immediately next door and repeat the entire process just described. Now we have two weights successfully nudged to error-reducing values. Proceeding stepwise in this fashion, through every one of the network's connections, produces a "new" network, new in having a slightly different connective configuration than the net with which we began, and an ever-so-slightly better performance at the output vector. Now we repeat this lengthy procedure with a second input-output pair, then a third, a fourth, and so on.

This sounds like a tedious undertaking, and it would be, if we had to do it ourselves. Fortunately, we can assign the entire business of vector presentation, error calculation, and repeated weight adjustments to a conventional serial computer, and then just stand back and watch the process unfold automatically. All of the input vectors in the training set are paired with their proprietary output vectors and stored in the computer's memory, and the computer is programmed to present each one to the student neural network, compute the error involved in each output, and adjust the weights according to the principles just outlined. After each presentation of each input-output pair, the computer nudges all of the network's weights to a slightly happier configuration.

We instruct the computer to keep repeating this procedure, for all of the input-output pairs in the training set, until the mean squared error of the student network's output performance is made as small as possible; until, that is, its performance on the training set has "topped out." Depending on the complexity of the network, this

can take hours, days, or even weeks of furious computing on the best available machines. But it regularly leaves us with a network that has genuinely *learned* the skill or transformational capacity in question.

Once this goal has been achieved, the training ends and the network's weights are frozen at their final values. We then have a network whose cognitive capacities and internal coding strategies we can begin to explore.

Performance in the Trained Network

Cottrell's face-recognition network topped out at an impressive level of performance. It achieved 100 percent accuracy, on the training set of images, with respect to faceness, gender, and the identity of the face presented. In itself, this is not necessarily impressive, since the network may just have "memorized" the finite set of input-output pairs presented to it during training, rather than acquired some truly general understanding of how to represent faces. A more severe and more relevant test occurs when we present the network with photos it has never seen before, that is, with various photos of the same people drawn from outside the training set. Here again the network comes through. It identified correctly 98 percent of novel photographs of the people encountered in its training set, missing the name and gender of only one female subject.

A second and even more severe test of its ability to generalize involved asking it to discriminate the faceness and gender of completely novel scenes and people. Here it was 100 percent correct on whether or not it was confronting a human face, and it was roughly 81 percent correct in its verdicts on the gender of the novel faces presented. (It got the male faces mostly correct, but showed a decided tendency to misclassify some of the female faces as male.)

A third and highly intriguing experiment tested the network's ability to recognize and identify a "familiar" person when one-fifth of the person's face was obscured by a horizontal bar across the input image (figure 3.6a). Surprisingly, the network's performance was hardly impaired at all. Subjects were identified correctly despite the obstruction, save in the one set of cases where the bar was placed so as to obscure each subject's forehead. For those inputs, performance fell to 71 percent correct, indicating that characteristic variations in hair position across the forehead must have

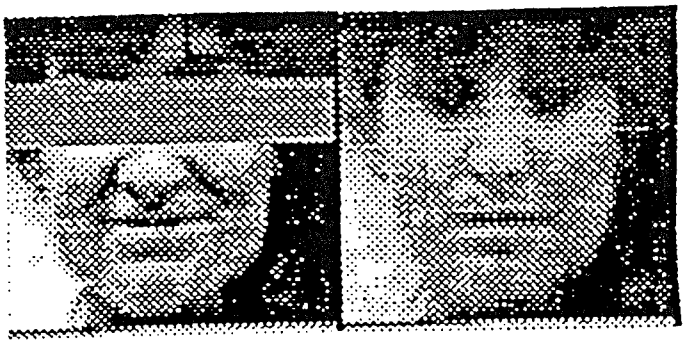

Figure 3.6 (a) An input image of a face one-fifth obscured by a bar. (b) How the representation at the middle layer of Cottrell's compression network fills in the obscured input area with coherently related facial features. (Thanks to Gary Cottrell.)

played a relatively large but far from exhaustive role in shaping the network's learned classification of individuals.

A fourth and final experiment trained a closely similar network to identify a number of typical emotional states, as they were feigned in the facial expressions of the same group of subjects. But we will postpone discussion of such acquired social-psychological skills until the chapter on *social* perception and action.

Internal Coding and Distributed Representations

How on earth does the trained network do all of this? What is going on inside it to make these surprising skills possible? It is all very well to be told that a zillion synaptic weights have been "suitably configured," but is there any more informative or illuminating description of what has happened to the network during training?

Yes, there is. For starters, let us focus our attention on the 80 cells at the second layer of the network (see again figure 3.4). We might suspect that, as in the cartoon example, the cells at this layer constitute an 80-dimensional face space, a space in which each discriminable face will occupy a unique position, a position specified by a unique pattern or vector of activation levels across that cell population. And that suspicion is entirely correct. As we are about to see, the representational trick used in so many other cases is hard at work here as well, not just in theory, but in fact.

What we now want to know is this: exactly *what* facial features are being coded by the cells at this level of the network? What effective coding strategy was gradually discovered by the network

during the relentless pressure of its training period? To put the same question in a third and final form, what is the *preferred stimulus*, at the input layer, of each of these many cells?

With an artificial network, as opposed to a living brain, we can obtain an absolutely determinate answer to this question, for each and every one of the 80 cells at issue. This is because the serial computer that conducted the network's training knows the exact value of the synaptic connection between any two cells in the entire network. In particular, it knows the pattern of connection weights onto each of the 80 face-dimension cells. As with the connection pattern onto the simple T-detection cell of figure 3.1, such connection-weight patterns uniquely determine the retinal input pattern to which a given face cell will give its maximum response. By reading out the training computer's memory—of the network's final configuration of weights for that cell—we can reconstruct the retinal input pattern that constitutes its preferred stimulus. Indeed, we can reconstruct it in the form of an image that we can look at ourselves.

The result of doing all this, for each of the 80 cells, yields a bit of a surprise. We might have expected each of these cells to become focused on some localized facial feature such as nose length, mouth width, eye separation, and so forth. But reconstructing the actual preferred stimuli of the 80 face cells reveals that the network settled into a coding strategy quite different from this.

Figure 3.7 reconstructs the preferred stimulus of six typical face cells from layer two of Cottrell's network. One can see immediately that each cell comprehends the *entire surface* of the input layer, and represents an entire facelike structure, rather than just an isolated facial feature of some sort or other. Neither do these preferred stimuli correspond to individual faces in the original training set. (There are 80 cells, recall, but only eleven distinct faces in the training set.) Rather, they seem to embody a variety of decidedly *holistic* features or dimensions of facehood, dimensions for which ordinary language has no adequate vocabulary. And yet, a given face presented at the input layer will variously activate each of these 80 holistic features, thus producing an activation vector at the second layer, a vector unique to that face. And different photographs of the same person presented to the input layer will produce essentially the same activation vector across the cells of layer two, thus allowing the output cells at layer three to identify that individual correctly.

Figure 3.7 Six of many *holons*: the preferred stimuli of some of the cells in layer two of the face-recognition network in figure 3.3. Note that each preferred pattern spans the entire input space. (Thanks to Gary Cottrell.)

Janet Metcalfe, Cottrell's co-worker on this and other networks, coined the term *holon* for a preferred stimulus or feature of this diffuse, input-spanning sort. As we will see, it is a coding strategy that networks quite regularly discover during training. This is presumably because it is an efficient and effective way of coding the information needed to solve the problem of finding the right input-output transformation. But it has a vitally important virtue beyond its efficiency: it helps to make the trained network functionally persistent against scattered cell damage and synaptic malfunction.

Since each pixel segment of the input image has some small effect on—that is to say, its information is *distributed* across—every cell at the second layer, and since every cell at the second layer contains at least some important information about the entire input layer, the scattered loss of cells and connections throughout the network leaves us with a network whose function is somewhat degraded, but still closely similar to its function in its undamaged state. With both the coded representations *and* their transformations being widely distributed across the entire network, there is no "bottleneck," neither representational nor transformational, whose isolated failure would bring the network suddenly to its knees.

Beyond this distributed wrinkle, we are still in familiar territory so far as coding goes. As with taste, color, and smell, neural representation here still consists in an activation vector across the representing population of neurons. What is changing in these increasingly complex examples is the size of the neuronal popu-

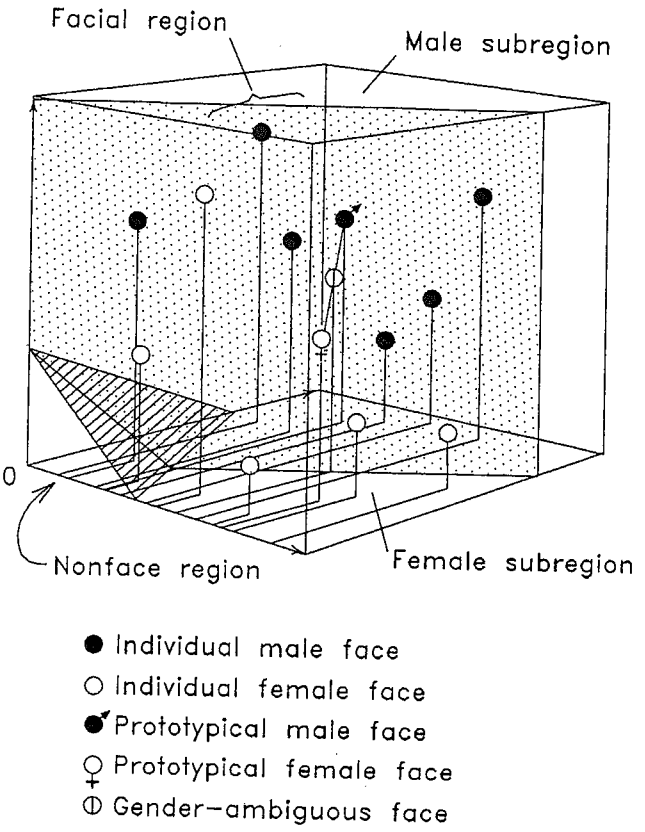

Figure 3.8 The hierarchy of learned partitions across the neuronal activation space of layer three (schematic).

lation, and thereby the dimensionality of the vectors involved. Most important, there is the increasingly holistic character of the features assigned to each dimension. Here also we begin to see something important and very welcome emerging: a hierarchical structure of recognizable *categories*.

The Emergence of Categories

Figure 3.8 attempts to portray the eighty-dimensional space of the cells in layer two of the trained network. It suppresses seventy-seven of them, of course, and settles for three of its typical dimensions so we can comprehend the point visually. Notice that, thanks to the network's training, this space now shows a primary partition into two regions: a large region in which all faces tend to be coded, and a smaller one close to the origin in which all nonfaces tend to be coded. This latter region is smaller because the cells at layer two respond hardly at all to any input that is not a face. The connection weights have been configured so that the dynamic range of

layer two is primarily spent on discriminating among faces. That is why the facial region is the larger of the two subspaces. If something isn't a face to begin with, it is hardly noticed, it barely even exists, so far as layer two is concerned. Notice also that there is no particular cutoff value, on any dimension, below which the coded subject must fail to be a face. A given face may produce a zero activation level in dimension 1, and yet still be coded as a face in virtue of having a sufficiently high value in dimensions 2 and 3. The vector {0, .5, .5}, for example, still codes a point well within the facial region.

Within that facial region is a secondary partition into male faces and female faces. These two regions are roughly equal in volume, reflecting the network's rough parity of discriminatory power within each class of faces. This is no surprise: it was trained on roughly equal numbers of male and female faces. Had the training set been biased in either direction, the resulting partition might well have been lopsided, as we will shortly see.

At the female subregion's "center of gravity" is the coding point for the prototypical female face. The prototypical male face occupies a complementary position next door. Scattered throughout these two subregions are the proprietary coding points for each of the several faces in the training set. These partitions across the activation space of the cells at layer two—these *categories*, for there is nothing else to call them—slowly emerged and stabilized during the course of the network's training. And it was to these partitions, and to the eleven "familiar face" points within them, that the cells at layer three slowly became tuned as well.

The appearance of these hierarchically divided regions provides us with a further way of describing and explaining the network's acquired skills of recognition and discrimination, a way beyond the austere vocabulary of synaptic connections. Not to put too fine an edge on it, what the network has developed during training is a family of rudimentary *concepts*, concepts that get variously activated by sensory inputs of the appropriate kind. This way of talking has yet to be fully justified, but you can already see the point of introducing it. The suggestion we are exploring is that the appearance of concepts in living cognitive creatures consists in the same sort of learned partitioning of neuronal activation spaces.

Return your attention to the partitions of figure 3.8. What is the significance of the dividing "walls" themselves? They certainly do not represent barriers of any sort. Rather, they represent planes of

indecision or uncertainty on the part of the network. An input image that produces a layer-two activation vector whose coding point lies on the basic face-nonface partition plane is an input image that the network is unable to discriminate as either a face or a nonface. Coding points at the boundaries between regions are coding points where the network, in effect, throws up its hands. Such coding points represent maximally *ambiguous* input images.

We can even generate them on purpose. Let us construct an input image that is the vector average of two images—the prototypical male face and the prototypical female face (see again the face of figure 2.6). When presented to the face network, it will produce an activation vector at layer two, a vector whose coding point lies midway between the two prototypical face points and precisely on the partition between the two subregions. Of course, the face of figure 2.6 is not the only face that is exactly ambiguous as between male and female—there are zillions of other possibilities elsewhere on the relevant partition—but it is an example quickly grasped.

And quickly experienced, too. If you wish to "hit" your own male-female partition across your own facial activation space, simply gaze again at figure 2.6. Individual reactions differ, it seems, as people's partitions are slightly idiosyncratic. But if your reaction is anything like mine, the face of figure 2.6 is highly ambiguous so far as its gender is concerned.

Is all of this really how humans recognize faces? Is the facial network at all realistic, biologically? Well, certainly not in the character of its third or output layer. That small population of cells is there only to provide the researchers with a convenient means of monitoring the network's performance. It is not intended to correspond to anything in the brain. But the gross anatomical outlines and the functional activity of the network connecting layers one and two may be a different matter.

On the anatomical issue, it must be said that the face-coding region in the human brain is at least five synaptic steps and five neuronal populations downstream from the retina, not one step, as with the crucial eighty-cell layer two in the artificial network. However, this need not be a significant contrast between reality and the model. The human visual system is concerned with many cognitive tasks in its first several layers, not just facial recognition. It must discriminate borders, shapes, typical objects, three-dimensional spatial relations, changes over time, physical movements, object trajectories, and more, all of it based on input vectors at the retina.

It would be no surprise if it takes such a busy network five, six, or even ten layers before it starts devoting proprietary processing resources to some special task such as face recognition.

There are many other known respects in which the artificial network is at best a clumsy and partial model of the intricate neuronal reality, and no doubt many more as yet unknown. But remember that we have reached the present stage of functional success in our artificial networks by trying to be as faithful to the neural reality as presently we can be. It is an empirical question whether deeper faithfulness will yield yet deeper success. At the very least, it is a question richly worth pursuing.

On the purely functional issue, we continue to find realistic aspects of the network's behavior beyond those already outlined. For example, networks of the kind at issue show what psychologists call "familiarity effects" in the profile of their discriminations. Alice O'Toole's group at the University of Texas, Dallas, trained facial-discrimination networks similar to Cottrell's on a training set that contained a relatively large number of Oriental faces plus a small number of Caucasian faces. The trained network showed a success level similar to Cottrell's, but it was less able to discriminate among, or to determine the sex of, novel faces that were Caucasian as opposed to Oriental. Although it performed just fine on Oriental faces, all Caucasian faces "looked pretty much the same" to it.

A second training experiment, this time with African faces in the majority over Oriental, yielded a network that performed fine on novel African faces, but poorly on Oriental faces. To the network, this time all of the Oriental faces looked pretty much the same. A third experiment, this time with a few African faces scattered among a large number of Caucasians, yielded a similar result. None of this is surprising. A network's acquired discriminatory capacities are maximally tuned to solving the recognition problems that it typically or most frequently encounters. If, during training, it encounters faces of type *A* much less frequently than faces of type *B*, then, if there are any systematic differences between faces of types *A* and *B*, the network will ultimately suffer a performance deficit where faces of type *A* are concerned.

The phenomenon is not only robust among the differently trained networks, it is familiar from human experience. Those who grew up in anything close to a monotonic racial environment have small but real discriminatory deficits for faces outside their racial group. Those who didn't, don't. Such deficits are of course avoidable, even

correctable, both in artificial networks and in humans. Simply pile on a training set that contains an equable distribution of the full range of human facial diversity. A healthy network adjusts its holons slightly, re-metrizes its face space accordingly, and the deficit disappears.

There is of course nothing essentially racial about this phenomenon. Had we trained the network on a preponderence of young faces over old ones, the network would have acquired a relative deficit in discriminating among old faces. Had we trained it on a preponderence of female over male faces, it would have acquired a relative deficit in discriminating among male faces. The basic point is that a common human failing appears to be a simple and explicable consequence of how neural networks partition their activation spaces in response to their ongoing experience.

Inductive Inference, Network Style

A related wrinkle in the profile of network learning yields a very advantageous property. Recall the ability of Cottrell's network to identify a familiar face hidden behind a horizontal bar. This ability to identify familiar faces correctly despite the loss of 20 percent of the information in the input vector illustrates a marvellous property of neural networks, a property with far-reaching consequences. This property is their capacity for what is called *vector completion.* Since the output layer identifies the input person correctly as Jane, it must be responding to a facial coding vector at the second layer that is very close to the proper coding point for Jane. At the least, that second-layer coding vector must be closer to Jane's point than to any other person's coding point, else the output layer could not have identified her correctly.

How is it that something very close to a complete Jane-vector at layer two can result from a 20 percent incomplete image vector of Jane at the input layer? The answer, in part, is that the information in the remaining 80 percent of the input image is still strictly sufficient to distinguish Jane's face, if not from *every* other possible face, at least from each of the other ten faces in the training set. Although there might exist other faces identical to Jane's face save for that portion of her face hidden behind the bar, certainly none of the other faces in the training set meet this condition.

The second part of the answer is that the network's second layer, having been trained primarily on faces, tends to fill in any empty

portion of an input facial image with generic facelike features, features roughly consistent, however, with the unobscured features of the input image. Figure 3.6b illustrates the informational content of layer two given the obscured image of a face (3.6a) as input. Compare that filled-in face to the face of the original and unobscured subject, the very first or upper-left female face in figure 3.3. The reconstruction isn't perfect, but it isn't half bad, either.

The third and most important part of the answer is that, during its training on the original eleven faces, the network has partitioned its activation space at layer two into a set of eleven "basins of attraction," each one centered on the prototypical coding point generated for each of the eleven faces. After all, it was not trained on all possible faces, only on these eleven. It was not asked to discriminate a million faces, just these eleven. The pressures thus exerted on the network during its training period forced it to become extremely sensitive to any and all input features that happen to point, even feebly, toward any one of exactly these eleven faces. This hypersensitivity regarding certain features of the input vector, and this forced concern with eleven specific individuals, means that the trained network has a strong tendency to "jump to conclusions" about the identity of an input face, even when the input is missing some standardly available information. To put the same point in another vocabulary, the network has acquired an especially strong tendency to activate, at layer two, some one or other of the eleven face-vectors to which it was so carefully trained, even when the input vector is partial or degraded. It tends to activate, at layer two, the specific facial vector that is most probable—*among the favored eleven*—given the "evidence" of the input vector received.

What we are observing here, in the phenomenon of vector completion at layers two and three, is a primitive form of *inductive inference*. Indeed, it may be that vector completion is the basic form that inductive inference takes in living creatures generally. We will address this issue in chapter 7 when we examine the nature of scientific reasoning. For now, let us just note that vector completion or inductive inference appears as a natural and inevitable cognitive phenomenon in the very simplest of neural networks.

With these points in place, we here leave the topic of face recognition, at least for now. I have spent a large portion of this chapter explaining how a specific network can achieve facial recognition

because I wished you to have a detailed understanding of how at least one multilayered network performs a familiar cognitive task. The face-recognition network is especially useful as an instructional example because it displays together, in a simple and intuitively accessible form, so many of the special features characteristic of neural networks and their style of operation. The lessons learned here are mostly repeated in the further examples explored in chapter 4. This will allow us to move through them rather more quickly.