Rapid Unsupervised Learning of Object Structural Descriptions

John E. Hummel and Jun Saiki
Department of Psychology
University of California, Los Angeles
jhummel@cognet.ucla.edu

Abstract

A single view of an unfamiliar object typically provides enough information about the object's shape to permit recognition from a wide range of new viewpoints. A recent model by Hummel and Biederman (1990, 1992) provides a partial account of this ability in terms of the activation of viewpoint invariant structural descriptions of (even unfamiliar) objects. We describe the Structural Description Encoder (SDE), a self-organizing feed-forward neural network that learns such descriptions in one or at most two exposures. Rapid, reliable learning results from the interactions among recruited and unrecruited units, whose response characteristics are differentiated through the use of dynamic thresholds and learning rates.

Introduction

For many objects, a single view provides enough information about the object's shape to permit recognition from a wide range of new viewpoints (e.g., Biederman & Gerhardstein, in press). This capacity, unremarkable from the standpoint of everyday experience, is mysterious from the perspective of theories of object recognition based on storing and matching multiple object views (e.g., Edelman & Weinshall, 1991; Intrator & Gold, 1993; Poggio & Edelman, 1990; Ullman & Basri, 1991, among many others). Such theories constitute the majority of computational models of object recognition.

Hummel and Biederman (1990, 1992) have proposed a neural network model of object recognition (based on Biederman's, 1987, theory of Recognition by Components) that accounts for part of this capacity. This model, *JIM*, recognizes objects in terms of structural descriptions specifying the objects' parts (simple volumetric primitives called *geons*) and the relations among them. Such descriptions are invariant under a wide range of viewpoints: they do not change with the size or position of the object's image on the retina, or the orientation in depth from which the object is depicted. Biederman and Cooper (1991; 1992a,b) have gathered a substantial amount of evidence supporting viewpoint invariant parts-based representations in human object recognition. JIM provides an explicit account of how a viewpoint invariant structural description can be derived from a line drawing of an object, but it does not address how these descriptions are encoded into memory for later use. This paper presents a neural network model of rapid, unsupervised learning of structural descriptions. The point of departure for this effort is Hummel and Biederman's JIM, so we briefly review that model.

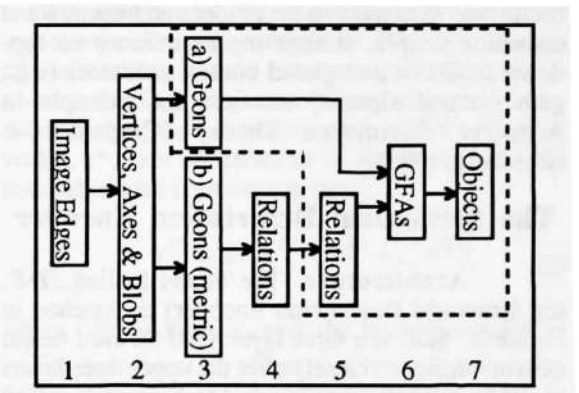

Figure 1. A thumbnail sketch of JIM's architecture. Layers 3a, 5, 6 and 7 are of primary interest here.

JIM is a seven layer feed forward neural network (Figure 1). Its first two layers represent a line drawing in terms of 2D image features (line segments, vertices, etc). Local computations in these layers segment line drawings into their parts by establishing synchrony of firing among units representing features of the same part. Units in layer 3 represent the geons in an image. A geon's categorical proerties (e.g., whether the cross section is straight or curved) and metric properties (e.g., size and location) are coded separately (layers 3a and 3b, respectively). Layers 4 and 5 use the geons' metric properties to calculate their relative positions, sizes, and orientations. Like the inputs to layer 3, the outputs of layers 3a and 5 are synchronized into sets: on a single time slice, the output of layers 3a and 5 form a *Geon Feature Assembly* (or *GFA*), an activation vector describing one geon in terms of its contrastive shape properties and its relations to the other geons in the image. The complete set of GFAs activated in response to an object is JIM's structural description of that object. Units in layer 6 (GFA units) respond to

specific GFAs. Units in layer 7 sum the outputs of GFA units over time, combining two or more GFAs to recognize a whole object.

The structural descriptions generated by JIM's first five layers suggest a natural account of how people can rapidly learn unfamiliar objects and later recognize them under a range of new views. JIM activates viewpoint invariant descriptions even for unfamiliar objects (i.e., for which no units are dedicated in layers 6 and 7), so encoding such a description -- by recruiting new units in layers 6 and 7 -- will naturally permit subsequent recognition from a variety of viewpoints.

The purpose of the model described here is to learn structural descriptions of the type generated by JIM. We have four specific goals in this effort. First, solve the old/new category problem: given a representation of an object, decide whether it is the first instance of a new category or an instance of a familiar category. Second, learn new descriptions rapidly (preferably, in one exposure) without catastrophically forgetting objects learned in the past. Third, learning must be unsupervised. And fourth, we wish to keep the model's architecture and operation simple, minimizing its reliance on top-down feedback and global control processes (e.g., gain control signals) as used for example in Adaptive Resonance Theory (Carpenter & Grossberg, 1987).

The Structural Description Encoder

Architecture The model (called *SDE*, for Structural Description Encoder) is sketched in Figure 2. SDE is a three layer, feed forward neural network designed closely after the upper three layers of JIM. SDE's first (input) layer consists of 18 units representing geon attributes and 7 units representing the relations among the geons. SDE takes GFAs as input; it is assumed that they are generated by a mechanism such as JIM's first five layers. Each object is represented by one GFA per geon. SDE's second layer consists of *GFA units* that self-organize in response to specific GFAs. The model's third layer consists of *Object units* that self-organize in response to conjunctions of active GFA units, that is, to complete objects. Connections between layers are feed-forward and excitatory, and within layers 2 and 3, units compete via lateral inhibition.

Learning Task Encoding a new object entails recruiting a small number of GFA units (perhaps only one) in response to each of an object's geons and one Object unit in response to the complete set of GFA units. This task is eased by the GFAs' invariance with viewpoint, which frees SDE to rapidly encode each new object without needing to maintain flexibility for generalization over viewpoints. Nonetheless, SDE's learning task is challenging, and must satisfy three constraints. First, the GFA-based representation is very dense, meaning that GFAs tend to be highly similar (as elaborated in the Simulations section), and some GFAs may even be subsets of others. Cohen and Grossberg (1987) and Marshall (1990) discuss the problem of learning embedded patterns in unsupervisded neural networks. The second constraint concerns the mechanics of deciding whether a pattern (GFA or whole object) is new or familiar. Unrecruited units must have an advantage over recruited units in responding to unfamiliar patterns, but recruited units must have the advantage in response to familiar patterns. And third, learning must occur in a single exposure, so unrecruited units must be capable of making large modifications to their incoming weights; but once recruited, units must remain stable to avoid catastrophic forgetting.

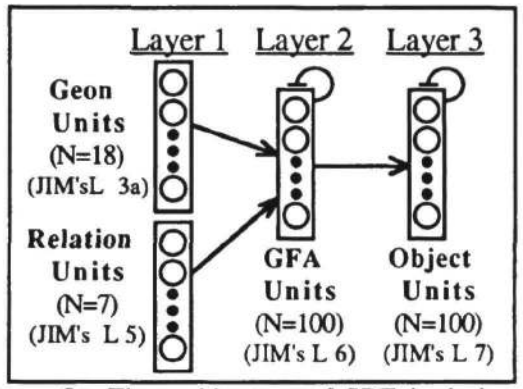

Figure 2. The architecture of SDE is designed closely after JIM's upper three layers.

Operation SDE is exposed to objects one at a time. The GFAs of each object are presented one at a time as patterns of activation (in the range 0..1) in layer 1. In response to each GFA, the GFA units compete via shunting inhibition, after which, those units whose activations exceed a threshold update their connections from the layer 1 units and pass activation to the Object units (layer 3). Object units sum their inputs over time and compete in a winner-take-all fashion. Only the winner updates its connections. As a simplifying assumption, Object units update their activations and connections only after all an object's GFAs have been presented. Initially, bottom-up weights are randomized (0..1 for GFA units, 0..0.05 for Object units), and all GFA and Object units start in an unrecruited state (described shortly).

The GFA Layer: The net input to GFA unit i is

$$N_i = (\sum_{j=1}^{n} w_{ij} I_j) \Big/ (\sum_{j=1}^{n} I_j^2), \quad (1)$$

where n is the number of units in layer 1 (n = 25), w_{ij} is the connection weight from unit j to unit i, and I_j is the activation of unit j in layer 1. If $N_i < \theta_i$ (unit i's threshold) then unit i 's activation, A_i, is set to 0; all other GFA units compete via shunting inhibition:

$$A_i = \theta_i N_i^3 \Big/ \sum_{j=1}^{m} \theta_j N_j^3, j \in (N_j > \theta_j), \quad (2)$$

where m is the number of GFA units (m = 100). Eq. 2 is applied for four iterations; after each, all activations below threshold are set to zero. Units update their weights by:

$$\Delta w_{ij} = A_i \alpha_i (I_j - w_{ij}), \quad (3)$$

where α_i is unit i 's learning rate.

Each unit changes its learning rate (α_i) and threshold (θ_i) as a function of learning. In its initial unrecruited state, a unit's threshold is low and its learning rate is high ($\theta_i = \theta_0 = 0.55$, and $\alpha_i = \alpha_0 = 0.9$). Each time a unit updates its weights, its learning rate decays:

$$\alpha_i^{t+1} = \alpha_i^t - (\alpha_i^t \varepsilon A_i^t), \quad (4)$$

where ε is a scaling parameter (ε = 0.8). Recruitedness is not all-or-none for GFA units. A unit's degree of recruitedness, is

$$\rho_i = (\alpha_0 - \alpha_i)/\alpha_0 . \quad (5)$$

As unit is recruited in response to a GFA (or an object), its thresholds rises:

$$\theta_i = \theta_0 + \beta e^{\gamma \sum_j \rho_j} \Big/ (1 + e^{\sigma(\rho_i - 0.5)}). \quad (6)$$

β (= 0.35) and γ (= -0.01) are scaling parameters, and σ (=-8) determines the steepness of the logistic function (the deominator is the recriprocal of the logistic function). The term in the numerator describes the threshold penalty for a fully recruited unit (i.e., ρ_i = 1.0) as a function of the sum of all ρ_i in the GFA layer: the greater the number of recruited units, the less the penalty on any single recruited unit. This convention serves to prevent units recruited early from responding to all patterns. The denominator of Eq. 6 determines the proportion of the threshold penalty applied to a unit as a function of its own degree of recruitedness.

The variable threshold plays a particularly important function in the competition between recruited and unrecruited units. Because θ is higher for recruited than unrecruited units, the former are more sensitive to deviations from their preferred inputs (as represented by their weights), and are therefore less likely to be activated above threshold by an unfamiliar pattern. However, given that it gets above threshold, a recruited unit has an advantage relative to the unrecruited units in the inhibitory competition (Eq. 2). This use of the threshold helps to ensure that only recruited units will respond to familiar patterns, and that at least some unrecruited units will respond to any unfamiliar pattern. Foldiak (1990) also uses variable thresholds in an unsupervised neural network, but the motivation and implementation of the variable threshold in his nework are very different from those described here.

The Object Layer: Object units are similar to GFA units except that Object units: (a) sum their inputs over time, (b) compete in a strictly winner-take-all fashion, and (c) become recruited in an all-or-none fashion. Object units sum their inputs by accumulating the activations of the GFA units in a vector, $\mathbf{v}^*$, over all iterations in which GFAs of the same object are presented as input:

$$\mathbf{v}^{k*} = \mathrm{N}(\mathbf{v}^{k1} + \mathbf{v}^{k2} + \ldots \mathbf{v}^{kz}), \quad (7)$$

where $\mathbf{v}^{k*}$ is the accumulated layer 2 activation vector for object k, $\mathbf{v}^{ki}$ is the layer 2 activation vector produced in response to the i th GFA in object k (GFA^{ki}), z is the number of GFAs belonging to object k, and N is a linear normalization. Object units compute their net inputs by Eq. 1, with $\mathrm{v_j}^{k*}$ (i.e., the j th element of $\mathbf{v}^{k*}$) substituted for I_j. Object unit thresholds are

$$\theta_i = \theta_0 + \rho_i \beta e^{\gamma(\sum_j \rho_j - 1)}, \quad (8)$$

where ρ_i is 1 for recruited units and 0 for unrecruited units. All other equations are the same as for GFA units. For Object units, θ_0 = 0.0275, α_0 = 0.9, and β, γ, and ε are 0.8, -0.01, and 0.8, respectively.

Simulations

SDE was trained to recognize the ten objects used by Hummel and Biederman (1992) to test JIM (illustrated graphically in Figure 3). Despite their dissimilarity in terms of pictorial

overlap, these objects are quite similar in terms of their GFA-based representations: the mean correlation between GFAs in the training set is 0.544, the maximum is 0.787, and the minimum is 0.356. Ideal performance would be as follows: upon the first exposure to an object, SDE would recruit a new and unique set of GFA and Object units; on subsequent presentations of the object, the same units would respond, and no new units would be recruited. Thus, SDE's performance should be independent of the order in which it is exposed to the objects. Objects were presented in four different learning schedules to assess SDE's speed of learning and robustness to presentation order.

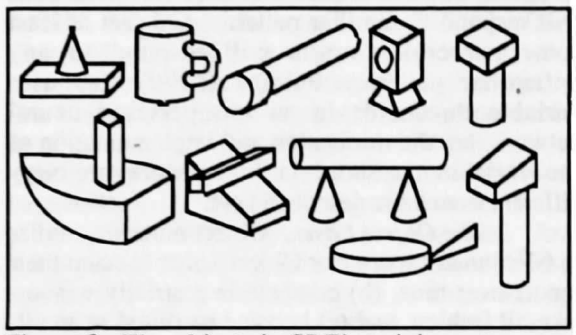

Figure 3. The objects in SDE's training set.

Simulation results are presented in terms of two types of errors: *split* errors and *merge* errors. A split error is when the model recruits different Object units in response to separate exposures to the same object, and reflects a failure to recognize a familiar object as familiar. Merge errors are when the same Object unit responds to two or more objects, reflecting the model's failure to treat different objects as different. Except where noted otherwise, all results are averages over 100 simulation runs. Error bars are shown on all graphs, but they are very small.

Random Order Within Blocks In the Random Order Within Blocks schedule, objects were presented in blocks of ten trials (one object per trial). In each block, objects were presented in a different random order. Trials proceeded as described previously. Each simulation was run for 24 blocks (240 trials). Figure 4 shows SDE's performance under this learning schedule.

SDE averaged about 50% split errors on block 2, indicating that it typically failed to recognize about half the objects to which it was exposed on the first block. (No split errors are shown for block 1 because they cannot occur until block 2 under this schedule.) For the other objects, learning occured in a single trial. By block 3, split errors are near zero, indicating that SDE typically learned all its objects within two exposures. SDE committed very few merge errors.

Completely Random Order In the Completely Random Order schedule, simulations were again run in 24 blocks of ten trials, but on each trial, every object had a 10% chance of being presented. Thus, a block is simply a series of ten trials, with no guarantee that all ten objects will appear within any given block. As shown in Figure 5, performance is very similar to performance under the previous schedule in that SDE leanred almost all its objects within two exposures (the difference in the shapes of the curves is attributable simply to when an object is expected to appear for the second time; the areas under the split error curve for this and the previous schedules are nearly the same). Once again, SDE almost never committed a merge error.

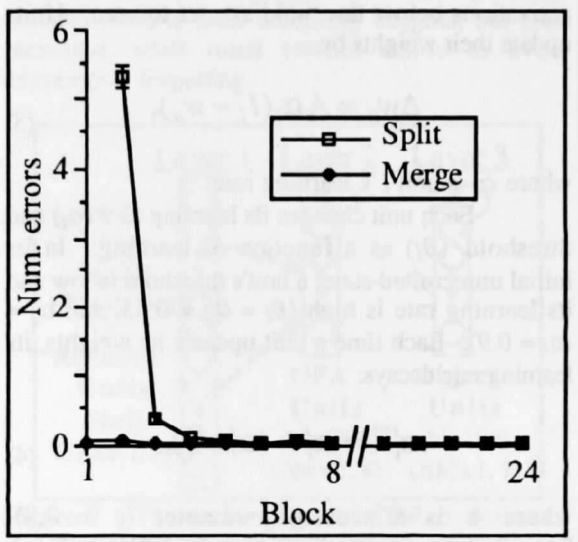

Figure 4. SDE's performance under the Random Order Within Blocks schedule, expressed as the mean number of errors produced per block (out ot 10 possible). The graph is truncated because the trend does not change between blocks 8 and 24.

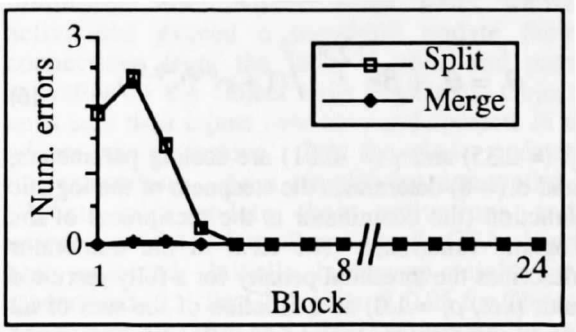

Figure 5. Performance under the Completely Random Order schedule. Under this schedule, split errors are possible on the first block.

Massed Trials Under the previous schedules, SDE received only one learning iteration per object before moving on to the next object, and although learning was rapid, it may improve with more time to encode an object on each exposure (anlogous to attending to a new object the first time one sees it). To test this hypothesis, SDE was tested under two Massed Trials schedules. In these schedules, SDE was given four uninterrupted learning trials (iterations) during each exposure to each object. Each block under this schedule thus reflects 40 trials (4 presentations of each object), so simulations were run for six blocks (240 trials). In the Random Order Massed schedule, objects were presented in a random order within each block. In the Fixed Order Massed schedule, objects appeared in the same order in every block within a 6-block run; this order was independently randomized on each of the 100 runs. The results of these simulations are shown in Figure 6. Both split and merge errors were nearly zero by the second block, indicating that SDE learned almost all its objects in a single four-iteration exposure. The random vs. fixed presentation order made virtually no difference in SDE's performance.

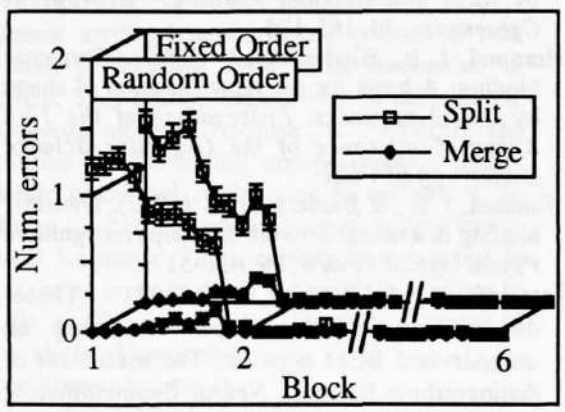

Figure 6. Performance under the Massed Trials learning schedules. Data are plotted in 4-trial (one object) increments.

Two-Phase The final learning schedule was designed as a strong test of SDE's ability to avoid catastrophic forgetting. In this Two-Phase schedule, five of the objects in the training set were randomly selected and presented for 12 learning blocks (first learning phase). In the second phase, the remainder of the objects were presented for learning. The critical question is whether objects learned in the first phase will have been forgotten after the second phase. Objects were assigned to phases randomly and independently on each simulation. Within phases, the schedule was Random Order Within Blocks. After the second phase of each run, SDE was tested for its ability to recognize that run's phase-1 objects. Figure 7 shows SDE's learning cruves under this schedule, averaged over 500 simulation runs.

Once again, SDE made virtually no merge errors. Interestingly, initial split errors were lower in the second phase of this schedule than in the first, suggesting that SDE learns objects faster when it already has experience with other objects. Most importantly, SDE made only 3 errors in 500 runs (0.6%) recognizing phase-1 objects after having learned the phase-2 objects.

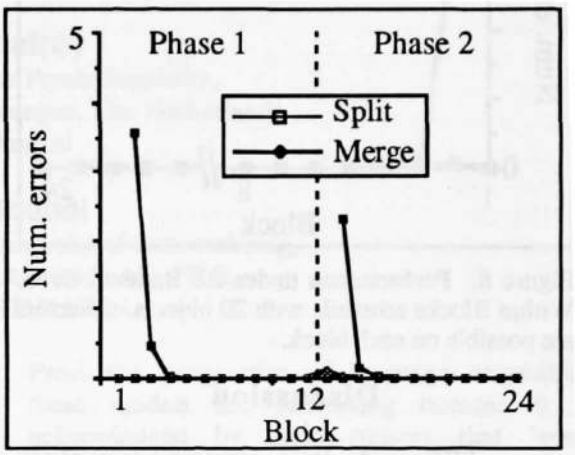

Figure 7. Performance under the Two-Phase schedule.

Noise Tolerance SDE does not require much capacity for generalization. Indeed, given the similarity of its input patterns, it needs to be sensitive to even a small number of large changes in GFAs, such as when a geon changes from having a straight cross section to having a curved one. However, high sensitivity to random noise is undesirable. We ran a series of transfer simulations in which noise was randomly added to or subtracted from the activation values in the training patterns. The likelihood of a change in a unit's activation was 0.8 and the magnitude of the change was normally distributed with a mean of 0.25. SDE made less than 1% errors recognizing the noisy objects, indicating that it readily generalizes over even a large number of small changes in its input vectors.

Scaling SDE performed very well on its 10 object training set, but it is important to know how it scales to larger training sets. To test this capacity, we ran it with a 20-object training set on the Random Order Within Blocks schedule. All parameters were exactly the same as in the previous simulations.

As shown in Figure 8, SDE made about the same absolute number of initial split errors as on the same schedule with only 10 objects (Figure 4). Therefore, in terms of its error rate, SDEs

initial performance was about twice as good with the 20-object training set as with the 10-object training set. This result strongly suggests that SDE's arcitecture and learning algorithm scale well with the number of objects it is required to learn.

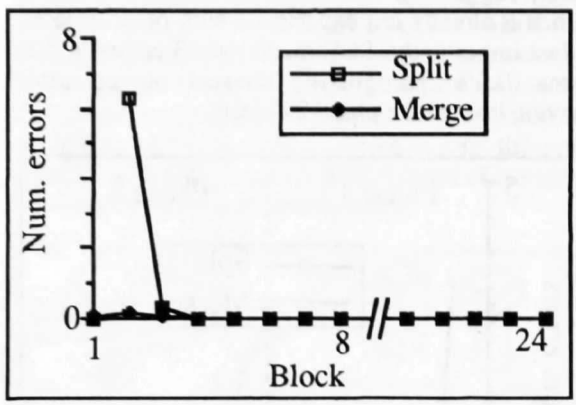

Figure 8. Performance under the Random Order Within Blocks schedule with 20 objects. 20 errors are possible on each block.

Discussion

SDE rapidly learns structural descriptions of the type generated by Hummel and Biederman's JIM model, and it's behavior is qualitatively robust across the various learning schedules investigated. Moreover, it tolerates noisy input patterns and scales well to larger training sets. SDE thus seems satisfactory as a first attempt to model rapid learning of object structural descriptions. It is worth emphasizing that SDE's success speaks to the efficiency of its GFA-based representation, which make rapid learning feasible by obviating the need for generalization over different viewpoints. As a neural network model of learning, SDE's most important contribution lies in the variable threshold, and in the distinction between recruited and unrecruited units. The intricacies of SDE's performance have yet to be investigated in detail, and systematic tests of human object learning will be important for evaluating the details of its behavior.

References

Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94, 2, 115-147.

Biederman, I. & Cooper, E. E. (1991). Priming contour deleted images: Evidence for intermediate representations in visual object recognition. *Cognitive Psychology, 23*, 393-419.

Biederman, I., & Cooper, E. E. (1992a). Evidence for complete translational and reflectional invariance in visual object priming. *Perception*, in press.

Biederman, I., & Cooper, E. E. (1992b). Scale Invariance in Visual Object Priming. *Journal of Experimental Psychology: Human Perception andPerformance, 18*, 121-133.

Biederman, I., & Gerhardstein, P. C. (in press). Recognizing Depth-Rotated Objects: Evidence and Conditions for 3D Viewpoint Invariance. Journal of Experimental Psychology: Human Perception and Performance.

Carpenter, G., & Grossberg, S. (1987). A massively parallel arcitecture for a self-organizing neural pattern recogntion machine. *Computer Vision, Graphics, and Image Processing, 37*, 54-115.

Cohen, M. & Grossberg, S. (1987). Masking fields: A massively parallel neural architecture for learning, recognizing, and predicting multiple groupings of patterned data. *Applied Optics, 26*, 1866-1891.

Edelman, S. & Weinshall, D. (1991). A self-organizing multiple-view representation of 3-D objects. *Biological Cybernetics, 64*, 209-219.

Foldiak, P. (1990). Forming sparse representations by local anti-Hebbian learning. *Biological Cybernetics, 36*, 165-170.

Hummel, J. E., Biederman, I. (1990). Dynamic binding: A basis for the representation of shape by neural networks. *Proceedings of the 12th Annual Conference of the Cognitive Science Society*, pp 614-621.

Hummel, J. E., & Biederman, I. (1992). Dynamic binding in a neural network for shape recognition. *Psychological Review, 99*, 480-517.

Intrator, N. & Gold, J. I. (1993). Three-dimensional object recognition using an unsupervised BCM network: The usefulness of distinguishing features. *Neural Computation, 5*, 61-74.

Marshall, J. A. (1990). A self-organizing scale-sensitive neural network. Proceedings of the International Joint Conference on Neural Networks, San Diego, CA, June, vol 3, 649-654.

Poggio, T. & Edemlan, S. (1990). A neural network that learns to recognize three-dimensional objects. *Nature, 317*, 314-319.

Ullman, S. & Basri, R. (1991). Recognition by liner combinations of models. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 13*, 992-1006.