Generation, Local Receptive Fields and Global Convergence Improve Perceptual Learning in Connectionist Networks

Vasant Honavar and Leonard Uhr
Computer Sciences Department
University of Wisconsin-Madison

Abstract

This paper presents and compares results for three types of connectionist networks on perceptual learning tasks:

[A] Multi-layered converging networks of neuron-like units, with each unit connected to a small randomly chosen subset of units in the adjacent layers, that learn by re-weighting of their links;

[B] Networks of neuron-like units structured into successively larger modules under brain-like topological constraints (such as layered, converging-diverging heterarchies and local receptive fields) that learn by re-weighting of their links;

[C] Networks with brain-like structures that learn by generation-discovery, which involves the growth of links and recruiting of units in addition to re-weighting of links.

Preliminary empirical results from simulation of these networks for perceptual recognition tasks show significant improvements in learning from using brain-like structures (e.g., local receptive fields, global convergence) over networks that lack such structure; further improvements in learning result from the use of generation in addition to reweighting of links.

Introduction

Connectionist networks are graphs of linked nodes. Each node is a simple neuron-like unit. Each link has a weight associated with it. The net input to a node is a weighted sum of the outputs of the nodes that fire into it. Each node applies some form of non-linear function (such as the threshold or the sigmoid) to its net input and sends the result to other nodes to which it is connected via its output links. The *receptive field* of a node is defined as the set of nodes that can directly fire into it.

It is easy to show that networks of threshold units are universal computing engines (McCulloch, 1943) in the sense that there exist (sufficiently large) networks of such units that can compute any function computable by a Turing machine or a system of Post Productions; but the problem of finding the necessary, sufficiently powerful, efficient and robust networks for perceptual recognition tasks remains, just as it does no matter how we try to embody intelligent processes.

A perceptual recognition system should be capable of interacting with constantly changing environments and, therefore, capable of learning. Learning can be viewed as a process of induction, constrained by the structure of the system as well as the input it receives from the environment. Given the complexity and the variety present in the real world, the number of possible structures relate the different inputs is extremely large: Given N inputs, each capable of taking V values, the number of possible structures relating them is V^N. This suggests that a perceptual learning system should be constrained, by its structures and processes, to learn the *meaningful* subset of relations between its inputs, given its limited resources, and the tasks it has to perform.

Learning in connectionist networks can involve modification of any of the following:

[1] Processing functions of the nodes (e.g., changes in the threshold or the output function),

[2] The weights associated with the links,

[3] The topology of the network (addition and deletion of links and nodes), and

[4] The learning rules themselves.

Most of the work on learning in connectionist networks to date has concentrated on [2]. Several algorithms for changing weights associated with the links are available (Hinton, 1987a). A learning scheme for [3] that employs a mechanism for growth of links and recruiting of nodes guided by regulatory mechanisms designed to discover minimally complex networks has been described in (Honavar, 1987; Honavar, 1988a).

Complexity Issues

Given the Turing equivalence of (sufficiently large) connectionist networks, the problem of building such networks for perceptual recognition tasks is reduced to one of discovering the design principles that yield economically feasible designs (for machine perception) and/or biologically plausible designs (for brain modeling). We briefly examine the complexity of perceptual recognition and list some observations on the physics of the environment and the structure of the brain that could potentially help us in deriving such design principles.

The complexity of recognition is $O(V^N)$ for an N-pixel image where each pixel can range through V values. This means that to handle the general recognition problem, including the worst case, a network needs at least V^N nodes, each linked (either directly or via intermediate nodes in layers or some other structure) to all nodes in the input

This work was partially supported by grants from the National Science Foundation, the Air Force Office of Scientific Research, and the University of Wisconsin Graduate School. A preliminary version of this paper appeared as a technical report (Honavar, 1988b).

retina. This of course is combinatorially explosive, and our real problem is the expected case, that is, recognition of real-world images. The human brain and its visual system is clearly capable of perception of real-world objects in real-time, yet it does rather poorly at the worst case e.g., telling apart two images that differ by a few randomly placed pixels. For the expected case E, the number of nodes needed is clearly within feasible bounds; otherwise nature could not have evolved brains capable of successful recognition.

If the structure of the human brain and the visual system is any indication, the necessary number of nodes, N_E, is still almost certainly extremely large, and the necessary topology G_E, of the network is far from random. A great deal is known about the human brain and the visual system (Peters, 1986; Uhr, 1986; Crick, 1986; Zeki, 1988; DeYoe, 1988; Livingstone, 1988). Neurons predominantly interact with near-neighbors and are organized into highly ordered structures (columns, hypercolumns, areas); Yet a great deal is unknown about how the neurons get allocated for computing specific functions, and how the detailed topology of the network of neurons emerges as a result of learning through constant exposure to the environment.

If the desired perceptual recognition abilities are to be attained by a connectionist network through re-weighting of its links alone, it must be initialized to contain *a sufficient number of appropriately linked nodes.* The only way to guarantee that this kind of network has enough nodes, each with the necessary links, is either to program them in, using a priori knowledge, or to make some guess as to N_E - and use a substantially larger number of nodes and links than that to be on the safe side. To handle the full vision problem the only completely safe thing to do would appear to be to use V^N nodes, each with N links - but this is impossibly large to actually implement.

Generation involving the addition of nodes and links enables a network to modify its topology, and appears to offer a way out of this dilemma. Given mechanisms to generate, the network can gradually grow, until the number of nodes approaches N_E and the network topology approaches G_E - whatever N_E and G_E may be. Thus there is no need to estimate E; this is done constructively by the network itself.

Rather than hope that some particular random or pre-programmed connectivity will work, or pay the excessive costs of complete connectivity, a system that generates can, under the implicit guidance of the environment's inputs and feedback, move toward sufficient connectivity. Generation works best hand-in-hand with the fine-tuning of functions provided by re-weighting of links. In addition, generation and re-weighting are probably best supplemented by mechanisms that break links when appropriate. Some of these issues, as well as a specific learning scheme combining generation and re-weighting, have been examined in (Honavar, 1988a).

Connectionist Network Structures Compared Experimentally

The multi-layered converging network structures studied include those that learn by re-weighting of their links (with no generation), using several types of connectivity - random, as well as restricted to near-neighbors, and those that learn by a combination of generation and re-weighting (where generation takes place within the constraints of near-neighbor connectivity). A summary of these network structures is given in figure 1.

Network structure	Receptive field	Generation	Built-in edges
[CP.R--]	Random	No	-
[CP.L--]	Local	No	No
[CP.L-E]	Local	No	Yes
[CP.LG-]	Local	Yes	No
[CP.LGE]	Local	Yes	Yes

Figure 1: *Summary of multi-layered, feed-forward, converging network structures; CP stands for the connectionist pyramids; R for random and L for local receptive fields; G for generation; E for built-in edge-detectors; a - in a given position indicates the absence of the corresponding network property; all use reweighting of links as a learning mechanism; only the last two use generation in addition to reweighting.*

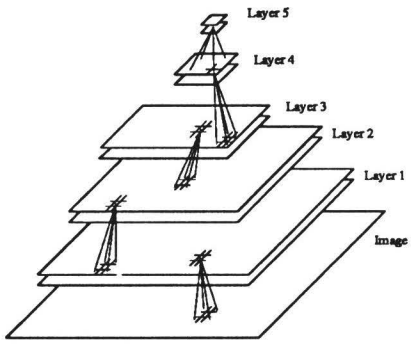

Figure 2: *A Converging pyramid-like structure: Each point in a layer has a cluster of nodes; Each node in a cluster computes a simple function over the outputs of nodes in the node-clusters in a small neighborhood in the layer below.*

Connectionist Networks That Learn by Re-Weighting

Several multi-layer, converging connectionist networks (using the same number of nodes and links in all the cases) were built, with the following structure:

Layer L contains 1/4th the number of node-clusters found in the adjacent layer $L-1$. Each node at layer L contains 4 times as many nodes per cluster as in the layer $L-1$. Each node in a node cluster at layer L receives input from 2-tuples of nodes drawn from 4 node clusters in layer $L-1$. In the current implementation, layer 2 is an exception in that each node in layer 2 receives input from 9 nodes (in a 3x3 window) in the input layer. This forms an overall pyramid-like converging-diverging structure (figure 2). In all the simulations described in this paper, the input layer (the retina) is a 32x32 array of pixels.

Three variants of the basic multi-layered, converging network described above were implemented:

[CP.L-E]

With local receptive fields preserving topographic

mapping between layers: each node in layer L is linked to nodes in the 4 node clusters spatially located directly below it in layer $L-1$; layer 1 contains 8 pre-wired edge detectors (these are simplified versions of the local spot and edge detectors found in the retina and primary visual area (V1) of living primate brains),

[CP.L--]

Same as [CP.L-E] above, but without the built-in edge detectors in layer 1, and

[CP.R--]

With random receptive fields: Each node in layer L is linked to nodes in 4 randomly chosen node clusters in layer $L-1$.

In all the simulations, 8 detectors (either pre-wired or learnable) were provided at layer 1. All the weights other than those corresponding to the built-in edge detectors were assigned randomly.

In all cases, learning involved re-weighting links as a function of the back-propagated error signal. Suppose a pattern class C_W is implied by the network with a weight W_W, and the pattern class indicated by the feedback, C_R is implied with a weight W_R; the amount of reweighting at the output layer is given by $(K \times (W_W - W_R))$ where K is a parameter related to the rate of learning. Our current implementation has K set equal to 0.25. This weight change is distributed equally among all the links firing into the node implying C_W. At internal nodes, the weight changes are computed in a similar fashion. This is similar in spirit to the *generalized delta rule* (Rumelhart, 1986).

Connectionist Networks That Learn By Generation and Discovery As Well As Re-weighting

Connectionist network structures that learn by generation and re-weighting of links and recruiting of new nodes from a pool of unused nodes were studied. The topological constraints on the network structure are the same as those present in [CP.L--] and [CP.L-E] described earlier. However, the networks that learn by generation as well as reweighting start with a pool of nodes and no pre-wired links. Generation grows new links and adds new nodes to the network from the pool of nodes as the network learns aided by feedback. The weights associated with the links are changed using the same reweighting mechanism as the one used in [CP.L--]. A particular implementation of generation and reweighting of this sort is described in (Honavar, 1988a).

Because generation does not violate the topological constraints of the layered, logarithmically converging organization as well as the local receptive fields, the networks that are discovered through generation and reweighting (e.g., [CP.LGE] and [CP.LG-]) are topologically similar to [CP.L--] and [CP.L-E]. But in contrast to [CP.L--], the number of nodes per node-cluster at a given layer in [CP.LG-] and [CP.LGE], or the connectivity between node clusters in adjacent layers, is not pre-programmed; it is determined dynamically through learning.

Runs were made with pre-wired edge-detectors in the first layer - [CP.LGE], and without any pre-wired nodes (i.e., having all the nodes added to the network as part of the learning process) - [CP.LG-]. In both these cases, the reweighting of nodes as a function of feedback proceeds according to the same reweighting rule as the one used in [CP.L--] and [CP.L-E]. In addition, the network occasionally generates a new node, when it determines this to be appropriate - on the basis of information provided by sub-structures that monitor the network's performance on each pattern class on which it is being trained. The design of these sub-structures is motivated by the need to discover the *simplest* networks capable of the desired accuracy of recognition. A particular implementation of such structures is explained in detail elsewhere (Honavar, 1988a).

The rationale behind the design is as follows: Continue to reweight existing links so long as the network's performance is improving. When it is observed that the network's performance has leveled off (before reaching the desired accuracy of recognition), generate a new transform. This is accomplished easily by a simple network of neuron-like units, using local computations that are performed incrementally following each training presentation (Honavar, 1988a).

Generation proceeds as follows: In the 1st layer, a 3-by-3 sub-array is extracted from the raw input image (this is done only when feedback indicates an error was made, and the history of the recent past indicates that performance is levelling off rather than improving. These 9 links fire into a new node placed directly *above* it in the next layer.

The extraction is got from a *busy* part of the input image, one where the network judges there may be useful information. The present simple system insists that a gradient be present, but potentially more powerful mechanisms that enable the system to evaluate a certain region (e.g., a 3x3 window) of the input for its information content, and their possible connectionist network implementations are being investigated.

In layers other than the 1st, extraction randomly links into a new node from 2 nodes that actively responded to the present (incorrectly identified) input image in the 2-by-2 of node-clusters directly below it in the previous layer.

Whenever a transform is generated, it is put into a node-cluster at that location, and also at every other location in that layer of the network. This makes translation-invariant recognition of patterns possible. All the links added to the network through generation get tuned through reweighting as a function of feedback.

Experimental Results

Several runs were made to compare multi-layered connectionist network structures ([CP.R--], [CP.L--], [CP.L-E], [CP.LG-] and [CP.LGE]). Simple 2-dimensional patterns such as letters of the alphabet (T, D, E) and simple objects (apple, cup, banana) were used for training the networks. The training and test sets were obtained by randomly dividing the set of drawings of each pattern provided by 3 different volunteers into two subsets. The drawings were made using the *Xgremlin* graphics utility on a Digital VAXstation-3200, in a 24x24 subarray of a 32x32 grid. A sample subset of patterns used is shown in figure 3. Figure 4 gives a summary of the pattern classes used in the runs,

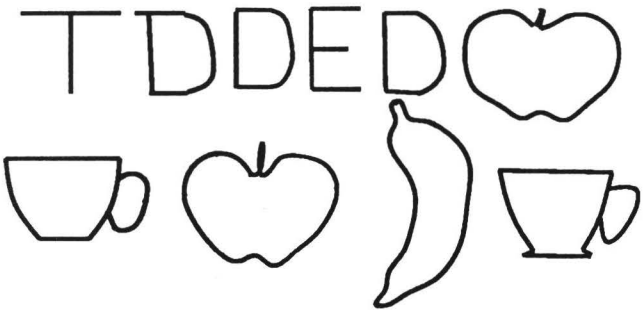

Figure 3: *Sample images used in the simulation of learning*

Pattern Set	# of classes	# of training instances per class	# of test instances per class
T, D, E	3	4	3
apple, cup, banana	3	4	4
T, D, E, apple, cup, banana	6	4	3

Figure 4: *Summary of pattern sets used in the experiments: the pattern sets were obtained from instances provided by 3 volunteers. Training and test instances for each class were obtained by randomly partitioning the set of instances for a given class into two subsets - one for training and the other for testing.*

i.e., (T, D, E), (apple, banana, cup) and the combined set (T, D, E, apple, banana, cup).

A *run* consists of several *epochs* of training interspersed with epochs of testing, repeated until the desired accuracy of recognition (currently set to 100%) is attained or the performance clearly levels off, as indicated by the learning curve. An *epoch* of training (or testing) involves cycling through the entire training set (or test set) once, in some arbitrary order. The runs for the structures [CP.R--], [CP.L-E] and [CP.L--] were made with several different percentages of possible connections (indicated next to the corresponding learning curves in figure 5), having fixed the number of nodes at each location in the first layer to 8, each with 9 connections.

In all cases, [CP.LGE] (pyramid convergence, locality, generation, built-in edge detectors) gave the best results, followed by [CP.LG-] (pyramid convergence, locality, no built-in edge detectors). These were both substantially better than the networks [CP.L-E] (pyramid convergence, locality, and built-in edge detectors), which in turn were substantially better than [CP.L--] (pyramid convergence, locality, no built-in edge detectors).

The figures 5 shows the results of these runs on the pattern set (T, D, E). The results with pattern sets (apple, cup, banana) were qualitatively similar in all the cases (the runs were slightly longer (took about 10% more epochs); about 10% more links were generated in [CP.LGE] and [CP.LG-]).

The networks [CP.R--] (random connectivity between layers, logarithmic convergence) failed to improve beyond

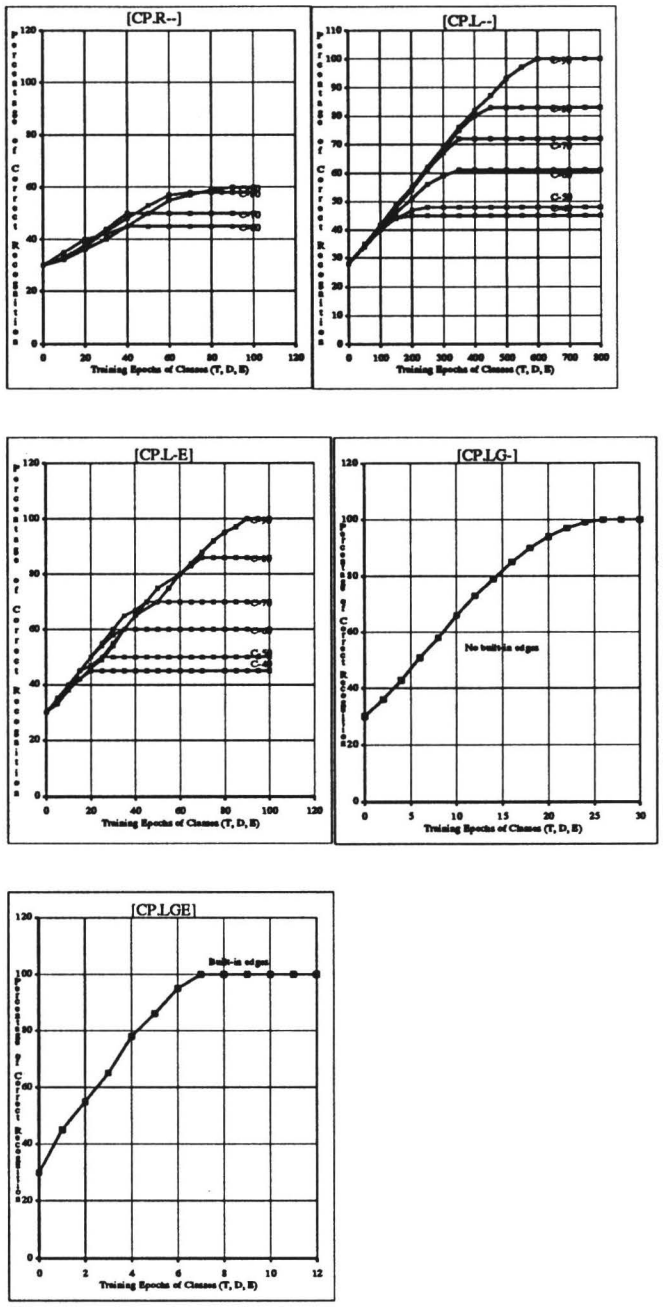

Figure 5: *Performance of networks on the recognition of test patterns as a function of the number of training epochs*

60% correct recognition given the same maximum number of connections that were used in [CP.L--] structures The networks [CP.L--] attained 100% accuracy of recognition with approximately 16×10^3 links, which were distributed equally between layers (1,2), (2,3), (3,4) and (4,5) in about 600 epochs of training, whereas the networks [CP.L-E] attained the same perfomance with the same network size, in about 90 epochs of training.

The network [CP.LG-] attained 100% accuracy of recognition in about 26 epochs with about 8×10^3 links (14 new transforms were generated and they were replicated at

each location in the corresponding layers). The network [CP.LGE] reached 100% correct recognition in about 8 epochs of training and at about 6×10^3 links (6 new transforms were generated and they were replicated at each location in the corresponding layers).

The runs were repeated for [CP.LG-] and [CP.LGE] with all 6 pattern classes (T, D, E, apple, banana, cup) and the results were qualitatively similar, but there were more generations (about twice as many) at the higher layers resulting in approximately 10×10^3 and 8×10^3 links respectively, and about twice as many epochs of training were needed for attaining 100% accuracy of recognition. The exact numbers reported here should not be given too much importance; however the results do suggest that other factors being equal, generation and local structure significantly improve learning, both in terms of the number of training epochs needed as well as the size of the networks necessary to attain the desired accuracy of recognition.

Discussion and Summary

Retinotopic mapping and near-neighbor connectivity exploit spatio-temporal contiguity present in the environment. Pyramid-like layered hierarchies enable the computation of complex functions as cascades and compounds of many simpler functions. Architectures embodying such topological constraints have been studied rather extensively for image processing and computer vision (Uhr, 1972; Burt, 1984; Rosenfeld, 1983; Uhr, 1987; Li, 1987). The results presented in this paper suggest that the incorporation of similar brain-like constraints on network structure can significantly reduce the complexity, and improve the learning speed, of connectionist networks that learn (as opposed to being carefully programmed) to perceive patterns. The initial choice of network connectivity is important. Random connectivity is unlikely to work in most practical problems. Similar conclusions were reached in an experiment to train a connectionist network to match random-dot stereograms (Qian, 1988).

Our results suggest that the addition of mechanisms that enable the network to grow new links as needed, under guidance from feedback, aided by network structures that enable it to monitor its own performance over time, yield further improvements in learning.

Intuition suggests that good system performance requires a proper match between the *entropy* of the source of external stimuli and the connectivity, both between the source and the system (Abu-Mostafa, 1988) as well as within the system itself. Generation relies on the environmental stimuli to develop the connectivity of the system. The resulting network is therefore likely to have a better match with the entropy of the environment than a network that starts out with a random subset of the possible connections and maintains its initial connectivity unchanged, so that learning can only adjust the weights associated with the links.

Generation in a multi-layered, converging network with local receptive fields ensures that successively more complex non-linear relations between features in the input encoding of patterns can be discovered at higher layers, to be assessed by the new transforms that are added. Thus the system is biased such that: learning of simpler features preceeds the learning of more complex relations; and successively more global relations are learned at successively higher layers. An examination of the transforms generated in the network simulations supports this intuition.

The extraction-generation programs described here do not discard bad transforms or place any limit on the number of nodes generated. Neither capability was needed for the test runs reported here, since these programs learned to recognize the pattern-sets they were tested on in relatively small number of training epochs. But to handle larger sets of more complex patterns, the ability to discard is almost certainly necessary - otherwise the network will get bogged down with many poor or worthless transforms.

There are a number of promising improvements to be made, including the addition of networks that make better assessments of potential generations, that learn to improve upon these assessments, that evaluate the generations for their usefulness for recognition, that discard poor generations to make room for new ones, that narrow and broaden the tolerance-threshold for matching, and that generate sets of alternate possible transforms that are placed in competition with one another. There are a number of other issues to be investigated, including the development of good sub-networks that realize functions for deciding whether to further re-weight or to generate, the optimal number of nodes in a node-cluster, and the desirability of putting the nodes within a cluster into direct competition.

The extent of generalization, i.e., building of meaningful internal representations by discarding uninteresting details, is an important property of connectionist networks that learn. More compact representations result from better generalization. There is reason to believe that the extent of generalization in connectionist networks is sensitive to the number of hidden units as well as the connectivity (Hinton, 1987b). If the hidden units (or connections) are too many, the network may generalize rather poorly; if they are too few, the network may never learn. Therefore, finding the optimal number of hidden units and/or weights is of interest. Generation and deletion of links can be seen in this context as providing mechanisms that dynamically determine the number of hidden units and connections needed in the network. Thus networks that generate only as needed may exhibit good generalization properties as well. Generation makes possible the linking up of an adequate number of units to solve a given problem; minimal generation favors the discovery of the smallest necessary number, and hence, better generalization. It would be interesting to examine this conjecture experimentally.

Sub-networks that maintain, update, and transmit as appropriate, information about the network's performance over time (e.g., a portion of the learning curve, used to trigger generation) offer several interesting mechanisms to influence learning that may be worth examining. Such structures may be used to alter learning strategies, rates of learning, thresholds of firing, each of which has an impact on the *plasticity* of the network. Future work will address some of these issues.

In connectionist networks that learn, feedback-guided reweighting of links in by small amounts effectively performs a *gradient descent* on a function that represents the error between the output desired and the output produced by the network so as to minimize that error. However, there is always a risk of getting caught in a local minimum, a shallow trough, or a valley in the error surface. Generation and discarding of transforms can be thought of as providing the network some means of climbing out of such local minima.

Most of the work on learning in connectionist networks has to date concentrated on reweighting schemes for modification of weights in a static topology. Recent anatomical and physiological studies suggest that learning may involve alteration of the number as well as the pattern of synaptic interconnections in the brain, in addition to changes in synaptic weights (Greenough, 1988; Honavar, 1989). The results presented in this paper suggest that there may be promising improvements to be realized using additional learning mechanisms that dynamically alter the network topology (e.g., generation), suitable constraints on the network structure for particular domains (such as local receptive fields and global convergence for vision) and regulatory mechanisms that alter the plasticity of the network, choose between different learning strategies, and so on. Extensive and systematic evaluations of networks incorporating one or more of these features for perceptual learning of pattern sets of varying degrees of complexity are needed in order to determine how they perform individually as well as collectively. The experiments and results discussed in this paper constitute at best, a preliminary exploration of only a few aspects of the problem. Work in progress is directed at examining some of these issues in greater detail.

References

Abu-Mostafa, Y., Connectivity and Entropy *Advances in Neural Information Processing systems* Morgan Kaufmann, San Mateo, CA (1988).

Burt, P. J., The pyramid as a structure for efficient computation, in *Multiresolution Image Processing and Analysis*, ed. A. Rosenfeld, Springer-Verlag, Berlin (1984).

Crick, F. H. C., and Asanuma, C., Certain aspects of the anatomy and physiology of the cerebral cortex, in *Parallel Distributed Processing, vol. 2: Psychological and Biological Models*, The MIT Press Cambridge, MA (1986).

DeYoe, E. A., and Van Essen, D. C., Concurrent processing streams in monkey visual cortex, *Trends in Neuroscience* **11-5** pp. 219-226 (1988).

Greenough, W. T., and Bailey, C. H., The anatomy of a memory: convergence of results across a diversity of tests *Trends in Neuroscience* **11** pp. 142-147 (1988).

Hinton, G. E., Connectionist learning procedures, *Technical report CMU-CS-87-115*, Computer Science Dept., Carnegie Mellon University, Pittsburgh, PA (1987a).

Hinton, G. E., Learning translation invariant recognition in a massively parallel network, in *PARLE: Parallel Architectures and Languages, Europe. Lecture Notes in Computer Science*, ed. G. Goos, and J. Hartmanis, Springer-Verlag, Berlin (1987b).

Honavar, V., and Uhr, L., Recognition Cones: A neuronal architecture for perception and learning, *Technical report #717*, Computer Sciences Dept., University of Wisconsin-Madison, Madison, WI (1987).

Honavar, V., and Uhr, L., A network of neuron-like units that learns to perceive by generation as well as reweighting of its links, in *Proceedings of the 1988 Connectionist Models Summer School*, ed. G. E. Hinton, T. J. Sejnowski, and D. S. Touretzky, Morgan Kaufmann, San Mateo, CA (1988a).

Honavar, V. and Uhr, L., Experimental results indicate that generation, local receptive fields and global convergence improve perceptual learning in connectionist networks, *Technical report #805*, Computer Sciences Dept., University of Wisconsin-Madison, Madison, WI (1988b).

Honavar, V., Perceptual development and learning: From behavioral, neurophysiological, and morphological evidence to computational models, *Technical report #818*, Computer Sciences Dept., University of Wisconsin-Madison, Madison, WI (1989).

Li, Z. N., and Uhr, L., Pyramid vision using key features to integrate image-driven bottom-up and model-driven top-down processes, *IEEE Transactions on Systems, Man, and Cybernetics* **17** pp. 250-263 (1987).

Livingstone, M., and Hubel, D., Segregation of form, color, movement, and depth: anatomy, physiology, and perception, *Science* **240** pp. 740-749 (1988).

McCulloch, W. S., and Pitts, W. H., A logical calculus of the ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics* **5** pp. 115-133 (1943).

Peters, A., and Jones, E. G., (ed) *Cerebral Cortex vol. 3: Visual Cortex*, Plenum, New York, NY (1986).

Qian, N., and Sejnowski, T. J., Learning to solve random-dot stereograms of dense and transparent surfaces with recurrent backpropagation, in *Proceedings of the 1988 Connectionist Models Summer School*, ed. G. E. Hinton, T. J. Sejnowski, and D. S. Touretzky, Morgan Kaufmann, San Mateo, CA (1988).

Rosenfeld, A., Pyramids: multiresolution image analysis, in *Proceedings of the Third Scandinavian Conference on Image Analysis*, (1983).

Rumelhart, D. E., Hinton, G. E., and Williams, R. J., Learning internal representations by error propagation, in *Parallel Distributed Processing vol. 1: Foundations*, The MIT Press, Cambridge, MA (1986).

Uhr, L., Layered recognition cone networks that preprocess, classify, and describe, *IEEE Transactions on Computers* **21** pp. 758-768 (1972).

Uhr, L., Toward a computational information processing model of object perception, *Technical report #651*, Computer Sciences Dept., University of Wisconsin-Madison, Madison, WI (1986).

Uhr, L., Highly parallel, hierarchical, recognition cone perceptual structures, in *Parallel Computer Vision*, ed. L. Uhr, Academic Press, New York, NY (1987).

Zeki, S. and Shipp, S., The functional logic of cortical connections, *Nature* **335** pp. 311-317 (1988).