Some Biases for Efficient Learning of Spatial, Temporal, and Spatio-Temporal Patterns

Vasant Honavar
Department of Computer Science
226 Atanasoff Hall
Iowa State University
Ames, IA 50011-1040

Abstract

This paper introduces and explores some representational biases for efficient learning of spatial, temporal, or spatio-temporal patterns in connectionist networks (CN) - massively parallel networks of simple computing elements. It examines learning mechanisms that constructively build up network structures that encode information from environmental stimuli at successively higher resolutions as needed for the tasks (e.g., perceptual recognition) that the network has to perform. Some simple examples are presented to illustrate the the basic structures and processes used in such networks to ensure the parsimony of learned representations - by guiding the system to focus its efforts at the minimal adequate resolution. Several extensions of the basic algorithm for efficient learning using multi-resolution representations of spatial, temporal, or spatio-temporal patterns are discussed.

1. Multi-Resolution Iconic Representations

Environmental stimuli (e.g., 2-dimensional visual images) typically contain features over multiple scales. Multi-resolution pattern encodings provide a basis for analyzing features in the environmental stimuli at different scales (Uhr, 1972; Rosenfeld, 1984; Dyer, 1987). This section introduces multi-resolution representations and their use in efficient learning of spatial, temporal, and spatio-temporal patterns.

Typically, a multi-resolution encoding scheme transforms the input (e.g., a 2-dimensional image) into a set of maps at successively coarser resolutions, each making explicit image features at a specific scale. An example of such a scheme is the *gaussian pyramid* in which successively higher levels encode blurred and sub-sampled versions of the immediately lower level (where the blurring may be applied to any *local* property of the stimulus e.g., intensity, color, texture, etc). If the stimulus is a 512x512 image, blurring with a 2-dimensional gaussian kernel

$$g(x,y) = \frac{1}{2\pi\gamma^2} e^{-\frac{x^2+y^2}{2\gamma^2}}$$

(γ controls the width of the kernel which is typically approximated by a 2-dimensional mask) and sub-sampling might

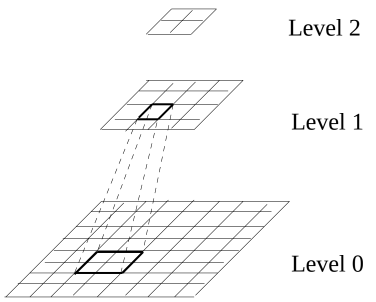

Figure 1: Multi-resolution Maps of a 2-Dimensional Pattern.

This research was partially supported by Iowa State University College of Liberal Arts and Sciences. This paper was published in the Proceedings of the International Joint Conference on Neural Networks, Beijing, China (1992).

be used to create a *gaussian pyramid* (Burt & Adelson, 1983) in which the successive levels have resolutions of 256x256, 128x128, 64x64, and so on (see below for details).

More complex multi-resolution representations (see below for details) might be constructed by applying cascades and compounds of transforms to the input image (Uhr, 1972; Hanson & Riseman, 1980; Honavar & Uhr, 1989). Such representations have been used with considerable success in a variety of image processing applications (Rosenfeld, 1984; Dyer, 1987). Given a suitable set of such transforms, one can construct systems which recognize objects in the input image (Uhr, 1972; Hanson & Riseman, 1980; Li & Uhr, 1987). Generative learning offers a way to discover the necessary set of transforms through feedback-guided learning (Honavar & Uhr, 1988; 1989).

2. Motivations for Learning from Multi-Resolution Representations

In spite of the obvious interest and considerable effort in the construction and analysis of multi-resolution stimulus encodings as evident from the literature in image processing and computer vision (Rosenfeld, 1984; Tanimoto & Klinger, 1980; Dyer, 1987), few attempts have been made to date to exploit the advantages of such representations to *learn* efficient encodings or internal representations of spatial, temporal, or spatio-temporal patterns.

The resolution at which a stimulus is encoded influences the content of perceptual experience. For example, perception of shape is sensitive to the resolution at which the contour of the object is represented. Often, features at relatively coarse resolutions are sufficient for making discriminations between different objects. These observations suggest that multi-resolution schemes can facilitate the learning of *parsimonious* internal representations of the environment.

3. Specification of Multi-Resolution Pyramids

An L-layer multi-resolution pyramid is shown schematically in figure 2. The nodes in layer 0 transduce (sense) the environmental input (e.g., a 2-dimensional color image at a given resolution $R_0 \times R_0$. It is straightforward to generalize this architecture when there are multiple input channels (e.g., one for each of the 3 primary colors). Let **X** denote the environmental stimulus transduced by the network at layer 0.

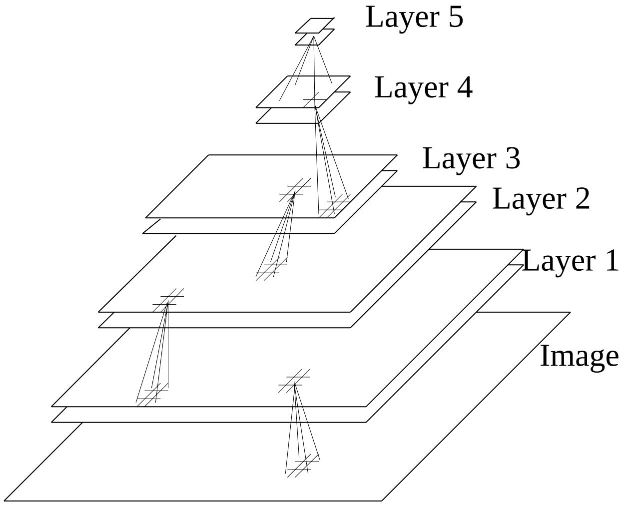

Figure 2: Locally Linked Logarithmically Converging Multi-Layer Network: Each location in a layer has a cluster of nodes; Each node in a cluster computes a simple function over the outputs of the nodes in clusters within a small window in the layer below, to which it is connected.

The spatial resolution at layer l is $R_l \times R_l$. The spatial resolution of the layers decreases (typically logarithmically as one moves from the input layer (layer 0) upwards. Thus, if the input layer has a resolution of, say, 512x512, assuming a uniform reduction in resolution between adjacent layers by a factor of 2 along each spatial dimension, the succeeding layers will have a resolution of 256x256, 128x128, 64x64, and so on.

Each layer l contains a cluster of nodes $\mathbf{N}_l(r,c)$ at each location (r,c). Let $n_{l,k}(r,c)$ be the kth node in the node cluster $\mathbf{N}_l(r,c)$; and K_l the number of nodes per cluster at each location in layer l.

$$\mathbf{N}_l(r,c) = \{n_{l,1}(r,c), n_{l,2}(r,c), .. n_{l,k}(r,c), .. n_{l,K_l}(r,c)\}$$

where

$$1 \le r \le R_l;\ 1 \le c \le R_l;\ k \in \{1,2, .., K_l\}$$

Let us denote the output of node $n_{l,k}(r,c)$ as a result of the input $\mathbf{X}$ to the network by $s^{\mathbf{X}}_{l,k}(r,c)$ (Typically $0 \le s^{\mathbf{X}}_{l,k}(r,c) \le 1$). Thus layer l in the multi-resolution pyramid has associated with it K_l 2-dimensional $(R_l \times R_l)$ arrays of outputs. Denoting the kth output array at layer l by $\mathbf{S}_{l,k}(\mathbf{X})$ we have (using a short-hand notation for a 2-dimensional matrix):

$$\mathbf{S}_{l,k}(\mathbf{X}) = \left[\ s^{\mathbf{X}}_{l,k}(r,c);\ \ 1 \le r \le R_l;\ 1 \le c \le R_l\ \right]$$

We denote the set of all outputs (as a result of the input $\mathbf{X}$ to the network) at layer l by $\mathbf{S}_l(\mathbf{X})$ where:

$$\mathbf{S}_l(\mathbf{X}) = \left[\mathbf{S}_{l,1}(\mathbf{X})\ \ \mathbf{S}_{l,2}(\mathbf{X})\ \ \cdot\ \cdot\ \cdot\ \ \mathbf{S}_{l,K_l}(\mathbf{X})\right]$$

At layer 0, the outputs are simply the corresponding input values (normalized and/or digitized if necessary). At higher layers the outputs of the nodes encode successively more complex abstractions of the input stimulus.

Each node $n_{l,k}(r,c)$ has associated with it, a simple *node function* $\mathbf{T}_{l,k}(r,c)$, a *convergence function* $\mathbf{Q}_{j,l,k}(r,c)$, and a *local mask* or *weight matrix* $\mathbf{W}_{j,l,k}(r,c)$ where j $(0 \le j < l)$ is the layer that provides input to the node $n_{l,k}(r,c)$. Different nodes in a single node cluster can receive inputs from different layers (but to keep things simple, all inputs to a given node are required to originate from a single layer).

Node function $\mathbf{T}_{l,k}$ = Gaussian match (radial basis function);
Convergence function $\mathbf{Q}_{j,l,k}$ = **Max**; e.g., if

$$\mathbf{F} = \begin{bmatrix} f(1,1) & f(1,2) & f(1,3) \\ f(2,1) & f(2,2) & f(2,3) \\ f(3,1) & f(2,3) & f(3,3) \end{bmatrix} = \left[f(p,q); 1 \le p \le 3;\ 1 \le q \le 3\right] = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

then $\mathbf{Max}(\mathbf{F}) = f(1,2) = 2$.
Let

$$u_{j,l,k}(a,b) = \sum_{p=-i}^{p=i} \sum_{q=-i}^{q=i} \sum_{r=1}^{r=K_j} \left(e_{j,l,k}(p,q,r) - s^{\mathbf{X}}_{j,r}(a+p, b+q)\right)^2$$

and

$$v_{j,l,k}(a,b) = e^{-\frac{u_{j,l,k}(a,b)}{\sigma_{j,k}{}^2}}$$

where $\sigma_{j,k}$ is a scaling or normalization factor.

$$s^{\mathbf{X}}_{l,k}(r,c) = \mathbf{Max}\left(\left[v_{j,l,k}(G_{l,j} \times r + g - \lfloor \tfrac{G_{l,j}}{2} \rfloor,\ G_{l,j} \times c + h - \lfloor \tfrac{G_{l,j}}{2} \rfloor);\ \ 0 \le g \le G_{l,j} - 1;\ 0 \le h \le G_{l,j} - 1\right]\right)$$

Figure 3A: Elementary network computations with a gaussian match node

---

To ensure translation-invariant perceptual recognition of 2-dimensional patterns, we assume that the same set of transforms is instantiated at each location (r,c) at layer l. In what follows, we will use $\mathbf{T}_{l,k}$, $\mathbf{Q}_{j,l,k}$, and $\mathbf{W}_{j,l,k}$ to denote the functions associated with the kth node in each of the node clusters at layer l irrespective of the coordinates of the particular node cluster in layer l. Thus we have:

$$\forall r \forall c \;\; \mathbf{T}_{l,k}(r,c) = \mathbf{T}_{l,k}$$
$$\forall r \forall c \;\; \mathbf{Q}_{j,l,k}(r,c) = \mathbf{Q}_{j,l,k}$$
$$\forall r \forall c \;\; \mathbf{W}_{j,l,k}(r,c) = \mathbf{W}_{j,l,k}$$

where $1 \le r \le R_l;\;\; 1 \le c \le R_l$.

$\mathbf{T}_{l,k}$ is one of several simple functions e.g., linear sum, linear sum and threshold, logistic, match, etc. (see below). Each $\mathbf{W}_{j,l,k}$ is a 2-dimensional $(z_{j,l,k} \times z_{j,l,k})$ matrix of K_j-dimensional vectors $\mathbf{E}_{j,l,k}(p,q)$ of stored values (e.g., weights) where

$$z_{j,l,k} = 2i+1;\;\; 0 \le i < R_j/2;\;\; -i \le p \le i;\;\; -i \le q \le i$$

For the sake of simplicity, let us assume $\forall k \;\; z_{j,l,k} = z_{l,k}$. Thus we have (using a short-hand notation for a 2-dimensional matrix):

$$\mathbf{W}_{j,l,k} = \left[\mathbf{E}_{j,l,k}(p,q);\;\; -i \le p \le +i;\;\; -i \le q \le +i \right]$$

where

$$\mathbf{E}_{j,l,k}(p,q) = [e_{j,l,k}((p,q),1) \;\; e_{j,l,k}((p,q),2) \;\; \cdot \;\cdot \;\cdot \;\; e_{j,l,k}((p,q),K_j)]$$

Each $\mathbf{Q}_{j,l,k}$ is a function (e.g., average, max, median, etc) defined over a 2-dimensional neighborhood at layer j. The size of the neighborhood is a function of spatial convergence, $G_{l,j}$ between layer j and layer l.

$$G_{l,j} = R_l/R_j$$

Figures 3A shows an example of elementary network computations involved in producing the output of the node $n_{l,k}(r,c)$ (i.e., the kth node in the node cluster at location (r,c) in layer l) for some typical choices of $\mathbf{T}_{l,k}$ and $\mathbf{Q}_{j,l,k}$ (To avoid unnecessarily cluttering the notation, we ignore the nodes at the borders of each layer; In practice, they can be treated as special cases - e.g., by padding additional rows of and columns of 0s as necessary).

Figure 3B illustrates a complete sequence of computations involved in producing a multi-resolution pyramid representation of the input **X**. In this case, a single transform (2-dimensional gaussian kernel) is applied uniformly at each location in each of the layers. In general, the system can employ multiple transforms at each location (each node in node-cluster applies a different transform) to construct multiple abstractions of the input at each layer.

More complex multi-resolution representations (e.g., recognition cones (Honavar & Uhr, 1989; Uhr, 1972; 1987)) might be constructed: Given an input pattern **X** transduced by the network at layer 0, each node $n_{l,k}(r,c)$ in a node cluster $N_l(r,c)$ at location (r,c) at layer l produces an output $s_{l,k}^{\mathbf{X}}(r,c)$. At layer l, there are K_l $R_l \times R_l$ 2-dimensional arrays of outputs where K_l is the number of nodes per cluster, and R_l is the resolution along each spatial dimension at layer l. Denoting the kth output array at layer l by $\mathbf{S}_{l,k}(\mathbf{X})$, we have

$$\mathbf{S}_{l,k}(\mathbf{X}) = [\; s_{l,k}^{\mathbf{X}}(r,c);\;\; 1 \le r \le R_l;\; 1 \le c \le R_l \;]$$

Each such array $S_{l,k}(\mathbf{X})$ is a representation of the stimulus at resolution $R_l \times R_l$. The set of all such stimulus representations at layer l is given by

$$\mathbf{S}_l(\mathbf{X}) = \left[\; \mathbf{S}_{l,1}(\mathbf{X}) \; \mathbf{S}_{l,2}(\mathbf{X}) \; . \; . \; \mathbf{S}_{l,K_l}(\mathbf{X}) \right]$$

These representations can be the result of potentially complex sequences of transformations (which may be built-in by design, or learned e.g., through generative learning (Honavar & Uhr, 1988; 1989; 1992a) of the input stimulus. We can denote the set of all such multi-resolution representations of a stimulus pattern **X** by in the network by **S**(**X**) where:

$$\mathbf{S}(\mathbf{X}) = \left[\; \mathbf{S}_0(\mathbf{X}) \; \mathbf{S}_1(\mathbf{X}) \; . \; . \; . \; \mathbf{S}_L(\mathbf{X}) \; \right]$$

For simplicity, let us assume: $\forall l \;\; K_l = 1;\; 1 \le l \le L$

Let the only allowed transformations be those between adjacent layers. Furthermore, let the same template function - a suitable $(2i+1)\times(2i+1)$ smoothing kernel (where i is a suitable small integer) be used at each location at each layer.

$$\mathbf{W} = \left[w(r,c);\; -i \le r \le i;\; -i \le c \le i \right]$$

For example, a 3×3 smoothing kernel $\mathbf{W}^3$ may be given by:

$$\mathbf{W}^3 = \begin{bmatrix} 0.05 & 0.05 & 0.05 \\ 0.05 & 0.60 & 0.05 \\ 0.05 & 0.05 & 0.05 \end{bmatrix}$$

Let the same node function $\mathbf{T}$ be used at each location in each layer.

$$\mathbf{T} = \text{subsampling function } \mathbf{Smpl}$$

Let $\mathbf{F} = \left[f(a,b);\; 1 \le a \le A;\; 1 \le b \le B \right]$ where $A\times B$ is the size of the neighborhood sampled.

Then $\mathbf{Smpl}\left(\mathbf{F}, (a^0, b^0) \right) = f(a^0, b^0)$

Thus the sub-sampling function picks a single element of $\mathbf{F}$ specified by the coordinates (a^0, b^0) where $1 \le a^0 \le A;\; 1 \le b^0 \le B$.

Let $u_j(a,b) = \sum_{p=-i}^{p=i} \sum_{q=-i}^{q=i} \left(w(p,q) \times s_j^{\mathbf{X}}(a+p, b+q) \right)$

This step corresponds to the smoothing operation applied at layer j. The smoothed version of $\mathbf{S}_j(\mathbf{X})$ is subsampled to obtain $\mathbf{S}_{j+1}(\mathbf{X})$. The neighborhood used for sub-sampling is a function of the convergence between adjacent layers. Assuming uniform convergence between layers, let

$$\forall j \;\; G_{j+1,j} = G^0$$

$$s_{j+1}^{\mathbf{X}}(r,c) = \mathbf{Smpl}\left(\left[u_j\left(G^0\times r + g - \left\lfloor \frac{G^0}{2} \right\rfloor,\; G^0\times c + h - \left\lfloor \frac{G^0}{2} \right\rfloor\right);\; 0 \le g \le G^0-1;\; 0 \le h \le G^0-1 \right], (0,0) \right)$$

Figure 3B: Elementary network computations in a multi-resolution pyramid.

4. Some Designs for Learning from Multi-Resolution Representations

As discussed earlier, the primary motivation for learning from multi-resolution representations is the need for the rapid construction of parsimonious internal representations of the environment which are necessary for the tasks (e.g., pattern recognition) that the system has to perform. This section outlines some designs for such systems. With each of these designs, the entire range of generative learning algorithms (Honavar & Uhr, 1992a) can be implemented and empirically evaluated. However, the focus of this paper is on generation through extraction of information-rich sub-patterns from the input (or successive abstractions of the input).

Feedback-guided Generative Learning

Generative learning structures enable a CN to extract potentially informative sub-patterns from the input and generate nodes that are optimally tuned to respond to the extracted pattern (Honavar & Uhr, 1988; 1992a). In brief, a generative learning system has to decide: when to generate a new node (as opposed to continued modification of weights associated with existing nodes); the sub-pattern of the input (or some abstraction of it) to be encoded by the generated node; the connectivity to be established between the output of the generated node and the rest of the network; and where (e.g., the layer at which) to initiate a generation.

Feedback-guided minimal generation with weight modification enables the system to initiate a generation from a misclassified sample pattern (or a sequence of such patterns) if the network's performance has not been improving at a sufficient rate over a sufficiently long sequence of training samples (Honavar & Uhr, 1988; 1989; 1992a; Gallant, 1990; Fahlman & Lebiere, 1990). Novelty-driven extraction provides the network with a novel (i.e., not currently encoded by the nodes in the network, and hence, potentially information-rich) sub-pattern from the input (or some abstraction of the input) to be encoded by the generated node (Honavar & Uhr, 1988; 1992a). These techniques contribute to the parsimony of internal representations of the environment discovered by the network through learning. Multi-resolution representations of spatial, temporal, or spatio-temporal patterns (e.g., gaussian pyramids, abstractions in recognition cones) are amenable to the use of additional biases that are designed to increase the efficiency of learning (measured by the effort (e.g., number of training samples) required for learning, or the memory utilized to store the learned representations).

Generative Learning from Simple Multi-Resolution Pyramids

It is useful to consider first, one of the simplest designs for learning from multi-resolution pyramids. Assume that the network is given the machinery to generate a multi-resolution (e.g., gaussian) pyramid from the input image. The general strategy used by the system is as follows: the system extracts (using generative learning processes discussed

```
while some performance criterion is not met do
    {
    sample stimulus X from environment at time t;
    construct multi-resolution abstractions S(X);
    compute the network output O for stimulus X;
    receive feedback F; compute error D(F,O);
    i = L;    /* L = top level in the multi-resolution pyramids */
    search = FALSE;
    if D(F,O) > ε   /* if error is intolerable */
        {
        .
        if criteria for initiating generation are met
            search = TRUE;
        while (i > 0 AND search == TRUE) do
            {
            attempt one or more novel extraction/s from S_i (X);
            if a novel sub-pattern E_i  is found in S_i (X)
                {
                generate a node at layer (i+1);
                initialize its weight vector to encode E_i ;
                replicate the node at each node cluster at layer (i+1);
                /* terminate the search for novel sub-pattern from S(X)* /
                search = FALSE;
                }
            else
                {
                i = i-1;   /* extend the search to the next higher resolution */
                }
            }
        }
    t = t+1;   /* look for the next sample pattern */
    }
```

Figure 4: An Algorithm for Efficient Learning from Simple Multi-Resolution Stimulus Pyramids

earlier) from the lowest resolution array in the gaussian pyramid a potentially informative novel pattern. The extracted pattern is encoded by the generated node which is linked to each of the output nodes (one for each distinct category). The search for novel extractions is extended to layer $l-1$ only when the system fails to find such an extraction at layer l (R_l, the resolution at level l is greater than R_{l-1}, the resolution at level $l-1$). Such an algorithm is outlined in figure 4. The key ideas are illustrated (using 1-dimensional patterns, for the sake of simplicity) in figure 5.

Generative Learning from Complex Multi-Resolution Pyramids

Unlike the simple gaussian pyramid considered above, multiple transforms might be applied at each layer of the pyramid to obtain multiple abstractions of the input at each layer. If the machinery for computing these abstractions is given a-priori, the system can proceed as before (using feedback-guided generative learning) to construct efficient internal representations needed for pattern recognition.

Unsupervised Generation of the Machinery for Computing Multi-Resolution Mappings

The network used to compute the multi-resolution maps of the input pattern can itself be learned using generation by extraction of novel sub-patterns from (small spatially local windows over) abstractions at the lower layers of the emerging pyramid (starting with the input). The nodes generated in this fashion are embedded into node clusters at the appropriate layer in the pyramid network. This process can proceed without supervision or external feedback (This has close parallels in the development of the visual system in mammals (Honavar, 1989)). Once the machinery for computing the multi-resolution maps is established in this manner, feedback-guided generation can take

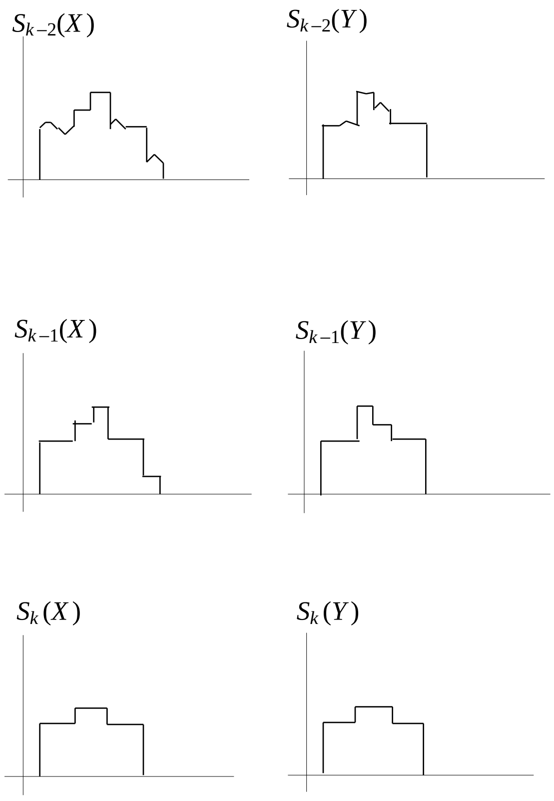

Figure 5: Multi-resolution maps of two 1-dimensional patterns **X** and **Y** are computed - say, by low-pass filtering and sub-sampling as in gaussian pyramids. Patterns **X** and **Y** are indistinguishable from each other at the spatial resolution R_k available at layer k. The system learns to distinguish between them when novel sub-patterns from layer $k-l$ (where $R_{k-1} > R_k$) are encoded through generative learning.

over.

Generation of the Machinery for Computing Multi-Resolution Mappings as Needed

Feedback-guided generative learning can be used to build up structures that compute multi-resolution representations of the input pattern which in turn are used to build up more such network structures in response to the system’s needs to perform a given a task (e.g., pattern recognition).

Information-Content of Patterns Extracted From Multi-Resolution Representations

The algorithm outlined above implicitly assumed that *novelty* of a pattern at a given resolution was a sufficient indicator of its information content. There are several schemes for increasing the information value and hence the utility of a generated node can be improved. Many such techniques (see Honavar & Uhr, 1992a for details) can be applied without modification to learning from multi-resolution representations: E.g., Instead of encoding a single novel pattern encountered, a small pool of nodes may be created with each node encoding a different novel pattern, and after a short exploratory period of evaluation, a small subset of that pool found most effective in improving the performance of the network may be incorporated into the network.

However, when extractions are obtained from small local windows (as opposed to the entire array at a given resolution) (Honavar & Uhr, 1988; 1991), additional regulatory control structures might be needed to guide the search at a higher resolution to find potentially useful sub-patterns that are not redundant (in terms of their utility for the task) with a sub-pattern already encoded in the network at a lower resolution.

Representational Bias in Multi-Resolution Pyramids

Multi-resolution pyramids incorporate fairly a strong representational bias: increasingly complex features over successively larger regions of the image are detected by nodes at successively higher layers of the network. This is accomplished by cascades and compounds of relatively local features detected at successive layers. The mappings across layers preserve the spatial or temporal structure implicitly encoded by the input image (this has close parallels to the retinotopic mapping that exists in the visual cortex). The mapping at each succcessive layer trades off some spatial resolution (and hence retinotopy) to encode knowledge of global structure in the image. The designs for learning from multi-resolution pyramids discussed above exploit the representational bias inherent in such structures to construct parsimonious representations necessary to perform a given task (e.g., spatio-temporal pattern recognition) - by directing the search for representations through successively higher resolution (and hence more detailed) maps of iconic input patterns. The effect of varying parameters that control this bias (convergence between layers, size of extractions, novelty threshold used in extraction, etc.) in such systems is currently under study.

6. Summary and Discussion

A system that has to learn to function in an unpredictable real-world environment can easily be overwhelmed by the amount of detail present in sensory data. Such a system ought to be able to discover the necessary detail (useful information) hidden in the data given the tasks that it has to perform. Feedback-guided minimal generation using novelty driven extraction of potentially useful sub-patterns from the input or its abstractions (Honavar & Uhr, 1988; 1989; 1992a) is a step in this direction. The techniques developed in this paper extend such a framework to multi-resolution representations of spatial, temporal, or spatio-temporal patterns. Formulation of similar inductive biases for learning efficient representations of three-dimensional objects is cutrently in progress (Honavar, 1992b).

Although we have used feedback-guided minimal generation by novelty-driven extraction of potentially useful sub-patterns as the primary learning mechanism for learning from multi-resolution representations of environmental stimuli, the approach can easily be extended to use a variety of other learning techniques including weight modification. In particular, multi-resolution pyramids of competitive learning networks (e.g., adaptive resonance models proposed in Carpenter & Grossberg, 1987) are currently under study. Developmental neurobiology of the visual system suggests further refinements of the representational and inductive biases introduced in this paper (Honavar, 1989). Development of computational models incorporating such biases and their empirical study is a topic of future research.

Although the examples used to illustrate the process of successive refinement of internal representations are extremely simple, the techniques used can potentially handle more complex, multi-dimensional, even multi-modal stimuli. Empirical studies with stimuli of varying degrees of complexity are in progress.

References

Burt, P. J., & Adelson, E. H. (1983). The laplacian image as a compact image code. *IEEE Transactions on Communications* 31 532-540.

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

Dyer, C. R. (1987). Multiscale image understanding. In L. Uhr (Ed.) *Parallel Computer Vision.* New York, NY: Academic Press.

Fahlman, S. E., & Lebiere, C. (1990). The cascade-correlation learning architecture. In *Advances in Neural Information Processing Systems (Vol. 2).* D. S. Touretzky (Ed.) San Mateo, CA: Morgan Kaufmann.

Gallant, S. I. (1990). Perceptron-based learning algorithms. *IEEE Transactions on Neural Networks* 1 179-.

Hanson, A. R., & Riseman, E. M. (1980). Processing cones: A computational structure for image analysis. In S. Tanimoto, & A. Klinger (Eds.) *Structured Computer Vision - Machine Perception Through Hierarchical Computational Structures.* New York, NY: Academic Press.

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

Honavar, V. (1989). Perceptual Development and Learning: From Behavioral, Neurophysiological, and Morphological Evidence to Computational Models. Technical report 818. Madison, WI: Computer Sciences Department.

Honavar, V., & Uhr, L. (1989). Brain-Structured connectionist networks that perceive and learn. *Connection Science - Journal of Neural Computing, Artificial Intelligence and Cognitive Research* 1 139-160.

Honavar, V., & Uhr, L. (1992a). Generative learning structures and processes for generalized connectionist networks. *Information Science* Special Issue on Neural Networks and Artificial Intelligence (To appear).

Honavar, V. (1992b). Learning Parsimonious Representations of Three-Dimensional Shapes. In *Proceedings of NATO Advanced Research Workshop on Mathematical Representations of Shape.* Netherlands (To appear).

Rosenfeld, A. (Ed.) (1984). *Multiresolution Image Processing and Analysis.* Berlin: Springer-Verlag.

Tanimoto, S., & Klinger, A. (Eds.) *Structured Computer Vision - Machine Perception Through Hierarchical Computational Structures.* New York, NY: Academic Press.

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