Cresceptron: A Self-Organizing Neural Network Which Grows Adaptively

John (Juyang) Weng, Narendra Ahuja, Thomas S. Huang
Beckman Institute, 405 N. Mathews Avenue
University of Illinois, Urbana, IL 61801 USA

Abstract

Cresceptron represents a new approach to neural networks. It uses a hierarchical framework to grow neural networks automatically, adaptively and incrementally through learning. At every level of the hierarchy, new concepts are detected automatically and the network grows by creating new neurons and synapses which memorize the new concepts and their context. The training samples are generalized to other perceptually equivalent items through hierarchical tolerance of deviation. The neural network recognizes the learned items and their variations by hierarchically associating the learned knowledge with the input. It segments the recognized items from the input through back tracking along the response paths.

1 Introduction

Backpropagation methods [5] can be used to iteratively modify the synaptic weights of a three-layer network so that the response of the network approaches the desired one. However, such a three-layer network trained by the backpropagation method has fundamental drawbacks. It is satisfactory only if the mapping from the feature space to the solution space is so simple that the local minima problem in the backpropagation iteration is negligible. Its high space complexity makes it intractable to deal with a large number of input variables. An addition of new items may require the network to be completely retrained.

We discuss the disadvantages of the popular three-layer networks as opposed to a network that has many layers (henceforth called hierarchical network). A challenging issue with the hierarchical network is, however, the modification of its synaptic weights during learning. We have developed an unsupervised learning framework based on the analysis of hierarchical concepts. Unlike conventional learning, the new learning method is incremental and thus growth becomes possible. New concepts are detected automatically, and the growth and interconnection of the network automatically adapt to the input information and the learned knowledge. Hierarchical tolerance of deviation makes it possible to handle many variations from a relatively small set of training samples. Since the learning is based on analysis at every level, the problem of local minima with the backpropagation methods has been avoided. It seems that this framework is closer to biological learning [2] (which appears free of local minima problem) than those that use backpropagation.

Among existing frameworks of neural network, a type called "Neocognitron" [1] probably most resembles the Cresceptron. However, there are fundamental differences between them. In the Neocognitron, the system designer specifies the breakdown of concepts, the concept each node represents, and the complete connections of the network. During the network development these designs must be modified by the designer in a trial-and-error fashion until an acceptable performance is achieved. In the Cresceptron, all these are done automatically to avoid intractable human intervention for complex tasks. Unlike the Neocognitron, if additional items are to be recognized, only incremental learning is needed. Moreover, in the Neocognitron, features are presented by the designer with slightly shifted positions to achieve the positional insensitivity. In the Cresceptron, a mechanism of hierarchical tolerance is built into the framework.

2 The Hierarchical Network

In this section, we investigate several important issues related to the complexity, sharing and flexibility of the network, as well as the advantages of the hierarchical network over three-layer ones.

0-7803-0559-0/92 $3.00

2.1 Complexity

A typical proof for the logical completeness of a three-layer network indicates its complexity. Suppose $\{a_i\}$ are input variables of a three-layer network. Each output x_i in the first layer corresponds to one side of a hyper plane: x_i is "high" if the corresponding weighted sum of $\{a_i\}$ exceeds a threshold. Each output y_i in the second layer takes the logical AND of $\{x_i\}$ and defines a region as the intersection of all the spaces defined by the hyper planes. Each output z_i in the third layer takes the logical OR of $\{y_i\}$ and defines a union of all the regions. Therefore, z_i can be expressed by a canonical logical expression of $\{x_i\}$ whose form is like:

$$z_i = x_1x_{12} + x_2x_{36}x_{38} + x_3x_4x_{23}x_{38}... \quad (1)$$

where AND is denoted by multiplication and OR is denoted by summation. Corresponding to a particular z_i, the number of hidden nodes in the second hidden layer equals the number of terms in the expression, and that of the first hidden layer equals the number of x_i's. The total number of hidden nodes equals the union of all such terms of $\{z_i\}$ with the same instances counted as one.

However, the formation of a complex concept is hierarchical. In the case of vision, an object is naturally described as a logical formation of its parts; each part is described in terms of primitives such as lines, curves, corners and regions; each primitive is described as a logical combination of lower level primitives such as edge segments. With this hierarchical description, the number of logical operations in the logical expression is much smaller than those in its canonical form. As an example, the expression

$$[(a_1 + b_1)(a_2 + b_2)...(a_n + b_n)][c_1d_1 + c_2d_2 + ...c_nd_n] \quad (2)$$

contains only $4n - 1$ logical operations. However, its canonical form contains $n2^n$ terms, each having 3 logical operations. (The same number of terms will result if the selected canonical form is of AND-of-OR instead.) Generally speaking, if an expression of n variables is converted into a canonical form, the number of terms is exponential in n. Although a three-layer network is logically complete, as is the canonical form in propositional calculus, its space complexity is very high compared to the hierarchical network.

2.2 Concept Sharing

Another advantage of hierarchical network is related to concept sharing. Each term in the canonical form represents a very low level grouping of concepts. Although different high level concepts may share the same hidden nodes in a three-layer network, such a sharing is very limited in scale and complexity.

In a hierarchical network, nodes at different levels represent concepts of different complexities. In the real world, many shapes have similar features, and many objects have similar parts. Therefore, a node in a hierarchical network can be shared by many high level nodes. Such sharing not only significantly reduces the number of nodes compared to the three-layer network, but also makes learning efficient.

2.3 Variation

In many applications, the neural network must be able to generalize. For example, it must be able to deal with some variations from the training samples. There are three possible ways to handle variations.

1. Invariant features. With this method, only features that are invariant with respect to the transformations are presented to the network. However, for complex real world problems, such invariant features, if they exist, are often too weak to discriminate between different classes.

2. Training with variation. The network is trained with typical variations in each class. For example, in face recognition, face images taken with several orientations need to be learned in order to recognize the face from any direction. However, in order to reduce the cost of training and the size of the network, the number of such variational samples should be kept small.

3. Structural tolerance. The network is designed to allow a certain amount of deviation from the training samples. The amount should be controlled to such a degree that it does not ruin the capability of discriminating different classes.

Our framework uses methods 2 and 3 above to accommodate variations, using the former for large variations and the latter for small ones. Through a hierarchical network, the structural tolerance can be allowed at every level. Different amounts of tolerance may be used at different levels of the network.

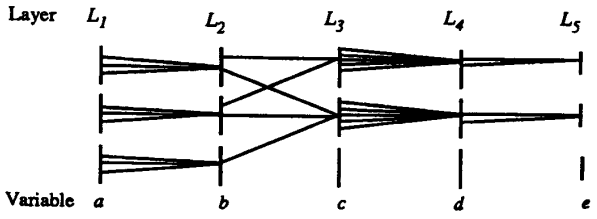

Figure 1: The schematic illustration of each module

3 Cresceptron

The term "Cresceptron" was coined from Latin *cresco* (grow) and *perceptio* (perception). The primary objective of the Cresceptron framework is to automatically handle manually intractable tasks: such as constructing a network that can recognize many objects from real world images. The Cresceptron uses a hierarchical structure, and the network adaptively and incrementally grows through learning. For recognition, the network should be largely translationally invariant, which is realized by using the same neuron at all the positions of each neural plane. Scale invariance is achieved through a multi-resolution representation with the framework of visual attention. Limited orientational invariance is obtained by the variation tolerance. Complete orientational invariance is not required here since the recognition should report also the orientation. Some studies have demonstrated that the human vision system does not have perfect invariance in either translation [4], scale [3], or orientation [6]. Since many principles discussed here are common with human perception, the Cresceptron can be modified for the use of other types of sensory data.

3.1 Visual Attention

For efficient examination of a complex image, saccadic scan and visual attention are incorporated into the system. To simulate different scales of visual attention, each input image is represented at different resolutions. There are two attention modes, manual and automatic. In the manual attention mode, the user selects an area of attention for recognition. In the automatic attention mode, the system automatically scans the entire image from coarse to fine resolutions and reports the result of recognition. The objective of visual attention is to scale an appropriate area in the image down to the size of the "fovea" of the neural network.

3.2 The Framework

The neural plane is the basic unit of the network. Each neural plane records the response of a particular type of neuron in the entire space of the fovea. The response pattern of a neural plane indicates the presence or absence of a particular concept (grouping of features).

The network consists of N modules, each corresponding to a concept level. The higher the level, the more complex the represented concepts are and the coarser the spatial resolution is.

The structure of the module is described in Figure 1. Each module has five layers, among which the first one is the input layer of the module and the last one is the output layer. Each layer has many planes, and the actual number depends on the automatic growing during learning. If every input plane of a module has $2^{k+1} \times 2^{k+1}$ grid points, every output plane of this module has $2^k \times 2^k$ grid points. The input layer of every module is also the output layer of its preceding module. In each module, the connections from the first layer L_1 to the third layer L_3 constitute a logical AND, and those from the third layer to the output layer perform a logical OR. Suppose $a(x, y, k)$ is the response at position (x, y) of plane k in layer L_1 and $c(x, y)$ is the response at (x, y) of a neural plane in layer L_3. The logical AND can be performed at position (x, y) by

$$c(x,y) = g(\sum_{k} \sum_{i=-t}^{t} \sum_{j=-t}^{t} w_{i,j,k} a(i+x, j+y, k)\ ) \tag{3}$$

where i and j run to cover the area centered at (x, y) in the neural plane ($t = 1$ in our implementation), $w_{i,j,k}$ is the synaptic weight, and g is a sigmoidal function whose saturation levels are determined by the

logical AND requirement. This expression requires many fan-in connections which can cause difficulties in hardware implementation. Thus, an intermediate layer L_2 is added whose response is denoted by $b(x, y, k)$:

$$b(x, y, k) = f(\sum_{i=-t}^{t} \sum_{j=-t}^{t} w_{i,j,k} a(i + x, j + y, k)) \tag{4}$$

where f is another sigmoidal function. Thus, $c(x, y)$ is modified as $c(x, y) = g(\sum_k b(x, y, k))$. The connections from layer L_3 to layer L_4 performs a template mapping:

$$d(x, y) = h(\sum_{i=-s}^{s} \sum_{j=-s}^{s} v_{i,j,k} c(i + x, j + y)) \tag{5}$$

where $v_{i,j,k}$ is the synaptic weight (element of the template) and h is a sigmoidal function ($s = 2$ in our implementation). From layer L_4 to layer L_5, the number of grid points in each neural plane is reduced by a factor of 4. This resolution reduction allows a slight deviation in the preceding response. Each grid point in layer L_5 receives the outputs from 4 grid points in the corresponding neural plane in L_4:

$$e(x, y) = q(\sum_{i=0}^{1} \sum_{j=0}^{1} d(2x + i, 2y + j)) \tag{6}$$

where q is a sigmoidal function which performs a logical OR. As long as the template matches at one of the four positions, $(2x, 2y)$, $(2x + 1, 2y)$ $(2x, 2y + 1)$, and $(2x + 1, 2y + 1)$, $e(x, y)$ is active in the output layer. The template matching in (5) ensures that only the patterns that are matched in the $(2s + 1) \times (2s + 1)$ window is allowed to pass to the OR operation that follows. Without this matching step, the OR operation can cause excessive tolerance, e.g., a diagonal line being recognized as a vertical line. As can be seen, the diameter of the receptive field of a node is increased by a factor of 2 at the output layer of each module.

This enables the network to hierarchically tolerate deviations from the learned samples. At the lowest level where the receptive field is very small, only pixel-level deviations are allowed. At a high level, the coarse grid implies that a large deviation is tolerated, which is desirable since the receptive field is large.

3.3 Detection of New Concepts

In the input layer of each module, the concept at a position is represented by the local active patterns at the corresponding positions in the neural planes. The concept is new if it has not been observed at any position. When an active pattern appears in the neural planes of the input layer, there exist three cases. In the first case, no neuron responds to this pattern. Thus, the concept is new. In the second case, some neurons respond, but they only respond to a part of the active input. In other words, no responding neuron connects to all the active neural planes. This implies that this concept is more complex than the concepts represented by the responding neurons, and therefore, it is also a new concept. Otherwise, the concept is not new.

3.4 Learning

The network is initialized to be empty. Then, a series of images are presented sequentially. From each image, the human operator selects the object to be learned from the input image by drawing a polyhedron that follows the outline of the object. Then, the network learns the object hierarchically. When a new concept is detected in a module, a new neuron with the synaptic connections is created. The input from the corresponding active neural planes is copied to the synaptic connections. The sigmoidal function determines an upper threshold such that if the same concept is presented, the output is saturated above. Since all the neurons at different positions of a neural plane are the same, only one neuron (with connections) needs to be created for each neural plane. Each training sample can cause many neurons to be created. Finally, if the sample is not recognized, a new node is created at the output layer of the highest level. A label that identifies the object is attached to this node so that later the node reports the label when it responds.

Over the entire network, knowledge sharing occurs naturally among different samples, since the same feature may appear in different samples. Therefore, the size increase of the network will gradually slow

down, while sufficient low and intermediate level knowledge has been learned to cover most of the training samples and thus further new nodes will be mostly needed for only high levels. This implies that this network can learn a large number of objects. When the space reaches the equipment limit, the network "forgets" less frequently used or very "old" knowledge by deleting the corresponding nodes and thus keeps its size manageable.

3.5 Decision Making

Once the network has been trained, it can be presented with new inputs for recognition. The result can be one of the following: (1) No output neuron responds: the new input is not recognized. (2) Only one output neuron responds: the input is recognized uniquely. (3) Two or more output neurons respond. The last case occurs when the input is similar to several learned samples. For example, the input contains a face that is taken at an orientation between those of the two learned samples. All the recognitions should be reported together with the response values as the confidence for further interpretation. For example, if the recognized objects indicate two different orientations of a face, a confidence weighted orientation sum can be used to approximately predict the actual viewing angle of the current face.

3.6 Segmentation

Once an object is recognized, the network can identify the location of the recognized object from the image. This is done by back tracking the response paths of the network from top level down to the lowest level. All the edges that have contributed to the recognition are marked in the input image. A closure can be easily computed from these marked edges to give the segmented region.

4 Experiments

The objective of this system is to locate and recognize general objects in real world scenes, including man-made objects such as buildings, furniture and machine parts, and natural objects such as trees, animals, and human faces. The final implementation of the system will consist of a front-end graphics-supported workstation and a fast massively parallel neurocomputer which is connected with the front end. The neurocomputer should be able to perform recognition tasks in real time.

For the theoretical and algorithmic development, this system is currently being simulated on a SUN SPARC workstation, and an interactive user interface has been developed which allows effortless training and examination of the network. Figure 2 shows the interface console of the system. The program has been written in C and its source code has about 5100 lines. The system digitizes video signals into (512×512)-pixel images or directly accepts digital images of resolution up to 512×512. The first version of the system uses the directional edges in the image as the input to the network. From each input image, the neural network accepts 8 edge images, each of which records the zero-crossings of the directional derivative of the Gaussian smoothed image along one of the 8 possible directions. The network has 7 modules, the fovea resolution is 64×64, and the maximum number of input connections of each node is 25.

A few dozens of sample images digitized from live TV programs have been used for training. A few of them are shown in Figure 3. A neural network has been automatically created through learning of these images. It has a total of 1367 output nodes at L_5 levels of various modules, and a total of 8143×9 synaptic connections. The system successfully recognizes the trained human figures in the sample images and can distinguish different objects. A few figures have been scaled slightly differently and be positioned at different places in the fovea, and they can still be recognized correctly. Figure 4 shows the results of segmentation from some of the recognized objects. While more and more instances have been learned, the system is able to recognize objects more precisely by the aforementioned confidence-weighted interpretation. Work is underway to use a much larger set of sample images and collect statistical data about the performance and the size of the network.

5 Conclusions

The representation of knowledge by a network hierarchy with many levels can reduce the space complexity of the neural network, allow knowledge sharing among different concepts, and enable the network to

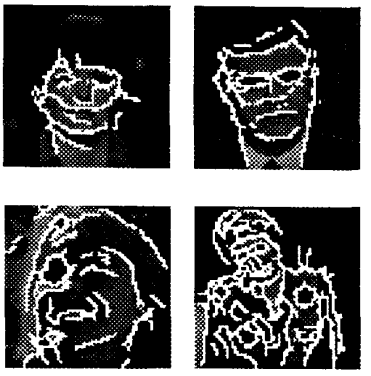

Figure 4: Some results of the segmentation for the recognized objects.

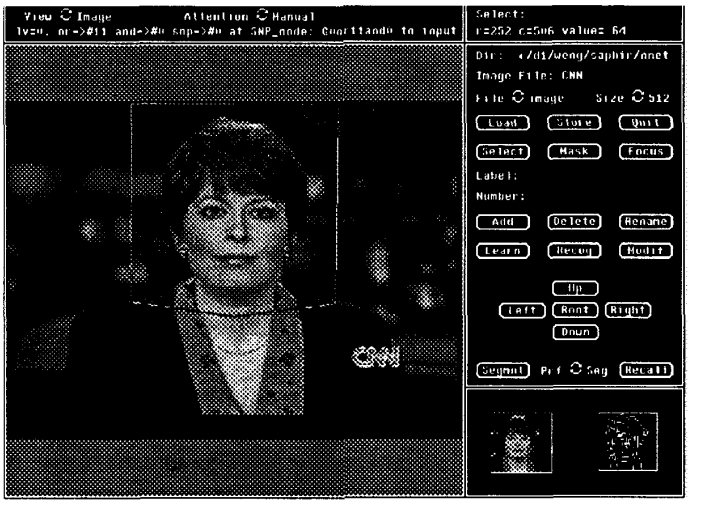

Figure 2: Interface console of the Cresceptron.

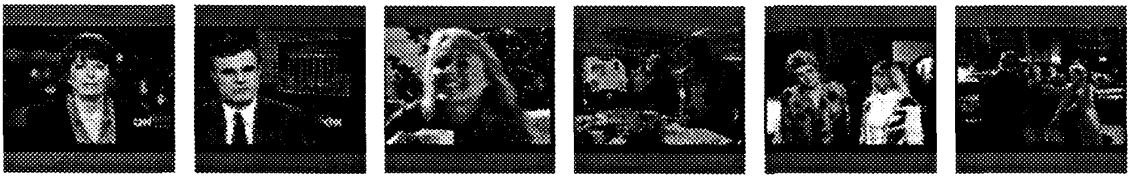

Figure 3: Some of the sample images for the training.

hierarchically tolerate deviations in the input. The Cresceptron is a framework for a self-organizing neural network by which new concepts are detected automatically and the network grows from empty by dynamicly generating new nodes that connect to the active neurons. Its topological interconnections fully adapt to the concepts being learned. Its built-in hierarchical tolerance enables the network to generalize the learned samples to other perceptually equivalent items. The connection of the network is local and thus facilitates hardware fabrication. Our experiments have shown that the network works properly and the results are promising. This work indicates a possible way of automatically creating an artificial intelligent machine by letting it grow, organize, and learn by itself.

References

[1] K. Fukushima, S. Miyake, and T. Ito, "Neocognitron: A neural network model for a mechanism of visual pattern recognition," *IEEE Trans. Systems, Man, Cybernetics*, vol. 13, no. 5, pp. 826-834, 1983.

[2] T. Kohonen, *Self-Organization and Associative Memory*, 2nd ed., Springer-Verlag, Berlin, 1988.

[3] P. A. Kolers, R. L. Duchnicky, and G. Sundstroem, "Size in visual processing of faces and words," *J. Exp. Psychol. Human Percept. Perform.*, vol. 11, pp. 726-751, 1985.

[4] T. A. Nazir and J. K. O'Regan, "Some results on translation invariance in the human visual system," *Spatial Vision*, vol. 5, no. 2, pp. 81-100, 1990.

[5] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in D. E. Rumelhart and J. L. McClelland (Eds.), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundations*, MIT Press, MA, 1986.

[6] P. Thompson, "Margaret Thatchar: a new illusion," *Perception*, vol. 9, pp. 483-484, 1980.