N88-16430

THE POWER OF NEURAL NETS

J. P. Ryan and B. H. Shah
Computer Science Department
University of Alabama in Huntsville
Huntsville, Alabama 35899

ABSTRACT

Neural net models work by simulating a collection of biological neurons and the interconnections between them. The learning abilities of their algorithms derive from ingenious ways to self-modify the connection weights. The specification of neural net models is done in terms of the characteristics of an individual node, the interconnection between the nodes, and the initial weights of interconnections and how they change. These models are based on the present understanding of how the biological neurons function. In this paper, we present implementation of the Hopfield net which is used in image processing type of applications where only partial information about the image may be available. Image-classification type algorithm of Hopfield and other learning algorithms, such as Boltzmann machine and back-propagation training algorithm have many vital applications in space.

1. INTRODUCTION

Neural net models are based on the present understanding of biological nervous systems since they offer many invaluable insights. Designing artificial neural nets to solve problems and studying real biological nets is changing the way we think about problems and lead to new insights and algorithmic improvements.

A neural network node as a model of a biological neuron is usually implemented as a non-linear processing element whose output is a non-linear function of input. Typically there are continuous input to and output from a node. An individual node is slow compared to modern digital circuitry, however, massive parallelism increases overall speed. An individual node weights ith input with a factor w_i and determines a weighted sum, S.

$$S = \sum_{i=1}^{N} w_i x_i$$

To each node is associated a quantity, theta, called internal offset, which determines a threshold above which the neuron represented by the node will fire. The quantity, alpha = S - theta, is passed through a non-linear function, f(alpha) to get the output. The three main types of non-linear functions are designated: (1) hard limiter (step function), (2) threshold logic element, and (3) sigmoidal non-linearity. Other more complex non-linearities (based on time dependencies, time integration, operations other than summation) are possible but they cause increased computation time.

2. SPECIFICATION OF A NEURAL NET MODEL

Following three quantities are generally required to specify a neural net model [3]:

A. Net topology: This includes interconnections among nodes and number of layers of nodes (one or two or more).

B. Node characteristics: This includes offsets of individual nodes and the type of non-linearity.

C. Learning rules: These are concerned with connection weights and include initial set of weights and how weights should be adapted during use to improve performance.

As previously mentioned, neural networks achieve high computation rates due to massive parallelism. Each computational unit is simple. Because of a large number of processing units we have a high degree of fault tolerance (or robustness). Damage to a few nodes will not significantly impair overall performance of the system. Another important feature is the adaptation of weights ("learning") based on new inputs. This enhances the robustness because even if there are minor variations in the characteristics of processing elements, the overall performance is maintained.

3. CLASSIFICATION PROBLEM

The classification problem can be stated as follows: determine which of M classes is most representative of an unknown static input pattern containing N input elements. Such problems are of common occurrence in many situations. Examples include: (a) speech recognizer, where input patterns are spectra of sounds and output classes are corresponding vowels or syllables; (b) image classifier, where input patterns are gray scale level of each pixel for a picture, and output classes are symbols identifying corresponding objects; and (c) spatial locator: where input patterns are omni-directed range measurements and output classes are identifications of sub-regions.

A neural net classifier is characterized by parallel computations and parallel input/output. N input elements are fed in parallel over N analog lines. Inputs may be bits or continuous over a range. The network first computes matching scores and then selects the maximum score and enhances it so that only one most likely class will be selected. Neural net classifier can be made adaptive to new classes or exemplars by using a learning algorithm that will modify the weights of connections as new classes are presented to the net. An implementation of a particular type of classifier is discussed in the next section.

4. THE HOPFIELD NET

The Hopfield network is designed for binary inputs. Examples of such situations include (a) pictures or images in terms of on-off pixels, and (b) ASCII representation of text with each character represented by 8-bits. Hopfield network applications are in associative memory and in solving optimization problems. In associative memory applications this network can locate the correct information from a supplied piece of partial information. When applying the Hopfield network the following limitations should be carefully considered.

(1) Spurious convergence: M should be small compared to N. If M is larger than about 15% of N, convergence may incorrectly occur to a pattern not matching any of the exemplars.

(2) Instability: A Hopfield net is said to be stable if upon using an exemplar as an input, the same exemplar is output. If too many bits are common between two patterns, the net becomes unstable.

We implemented the Hopfield network in Lisp on Texas Instruments Explorer which is a Lisp architecture computer. A run was made using two exemplar patterns with 171 pixels represented as *'s and -'s which are converted into +1 and -1 by the program, respectively. When the network is presented with a test input pattern with some of the pixels changed, it responds with the number of the exemplar which comes closest.

Another set of interesting runs was made with M = 2 exemplars and N = 10 pixels. We see that when only one pixel is different from the exemplars in the input pattern, the network does come up with the correct answers for the matching exemplar pattern. The algorithm converges to the correct exemplar pattern. The network was then presented with two input patterns which differed from both the exemplars in exactly the same number of pixels, namely, 5 pixels, half of all the pixels in each exemplar. The network does not converge to any of the exemplars, as would be expected.

We have also demonstrated what may be termed the "soldier's helmet" phenomenon. The network is presented with two different input patterns with only the first two pixels matching with either one of the exemplars. The network identifies the correct match and, in addition, the convergence is to the appropriate exemplar. This run dramatically illustrates the ability of the Hopfield network to reconstruct the whole picture from a partial one.

5. PLACE-FIELD AND GOAL LOCATION MODELS FOR A ROBOT

5.1. PLACE-FIELD MODEL

This model is based on the behavior of place-field cells of the hippocampus (one of two ridges along lateral ventricle of the brain) of a rat. Place-field cells fire at their maximum rate only when the animal is at a particular location relative to a set of landmarks. Such locations are called place fields. Zipser [5] developed a computational model to relate the configuration of landmarks to the location, size, and shape of place fields. The inputs to the model consist of configuration of landmarks in the environment together with the location of the observer. The output from the model represents the activity of a place-field neural unit in the observer's brain. A set of simple objects is used as place cues and the size of retinal images is indication of location. The model uses this information to provide quantitative predictions of how the shape and location of place fields change when the size, tilt, or location of these objects is changed.

The place-field neural model is designed for pattern recognition. It fires at maximum rate when the observer is at a desired location. A stored representation of a scene is compared to a representation of the current scene. Closer the viewer is to the stored scene, the better is the match. In determining the closeness, it is sufficient to use only a few discrete objects rather than the entire scene. A point P in two dimensional space can be uniquely located by its distance from three landmarks a, b, and c. It can be shown that in three dimensional space a point can be uniquely located by its distance from four landmarks.

When the robot is at a location P, the representations of the landmarks a, b, and c including the distance to P are recorded in some way in the memory. Upon return to P, the robot's sensory system can now generate a set of current distances. If these representations, other than their distance components, are not affected too much by the viewing postion, then the current representation of each object will differ from its stored representation only to the degree that the object's current distance from the observer differs from its distance to P. A neuron whose output is a summated measure of the similarity between these current and stored representations for each landmark will have the properties of a place-field unit.

5.2. CURRENT APPROACH

In contrast to the place location approach described above, we will not make the assumption of a "landmark recognizer" or any other sophisticated pattern recognition devices. Rather, our method will be dependent only upon the range sensor information.

The robot is assumed to operate in a two-dimensional region and to have range sensor detectors fixed in many directions from its center. The robot will travel translationally only -- so that it is always oriented towards a specific direction and there is no rotation. Many of these assumptions can be relaxed, as more sophistication is added to the model. The object of the robot is to explore the two-dimensional region to which it is confined. After sufficient exploration, it is able to navigate between any two points. As mentioned before the robot will rely only on range sensor data and will be unable to distinguish the angle at which it is approaching an obstacle. Movement of obstacles will be allowed and the robot will expected to navigate in a reasonable manner. So as not to become myopically confused by the obstacles and boundaries of the region, the robot will have a buffer distance always separating it from any other objects or boundary. The robot sensors are assumed to be of unlimited distance.

Navigation mode of the robot will start by moving from a "corner" in the direction of one of its sensors. The robot will assigned a predefined rectangular subregion, R, which is subdivided by the robot for exploration. The key factor in determining the "acceptability" of a subregion is whether the sensor vector varies continuously. An unsatisfactory subregion will be further subdivided into smaller regions so that the sensor vector is continuous. The robot gathers sensor data and data on change in position from each successive rectangular subregion.

In the navigational mode, the robot will have to periodically back up in order to insure that its connection weights are sufficiently tuned to discriminate one subregion from another. This is a highly unsupervised type of learning scenario, so that only certain types of neural net models would be acceptable. The work on learning neural nets [2], namely, Boltzmann machine [1,2], the competitive neural net, and sigma-pi neural nets [4] is being continued.

REFERENCES

[1] S. E. Fahlman and G. E. Hinton, "Connectionist architectures for artificial intelligence", IEEE Computer, pp. 100-109, January 1987.

[2] G. E. Hinton and T. J. Sejnowski, "Learning and relearning in Boltzmann machines", in "Parallel distributed processing: Explorations in the microstructure of cognition, Vol. 1, Foundations", edited by D. E. Rumelhart and J. L. McClelland, M. I. T. Press, pp. 282-317, 1986.

[3] R. P. Lippmann, "An introduction to computing with neural nets", IEEE ASSP Magazine, pp. 4-22, April 1987.

[4] T. Maxwell, C. L. Giles, and Y. C. Lee, "Generalization in neural networks: the contiguity problem", to be publiched in the "Proceedings of the IEEE International Conference on Neural Networks", San Diego, Calif., June 1987.

[5] D. Zipser, "Biologically plausible models of place recognition and goal location", in "Parallel distributed processing: Explorations in the microstructure of cognition, Vol. 2, Psychological and biological models", edited by J. L. McClelland and D. E. Rumelhart, M. I. T. Press, pp. 432-470, 1986.