CRH SSD
NOVEMBER 1990

CENTRAL REGION TECHNICAL ATTACHMENT 90-39

A NEURAL NETWORK FORECAST DEMONSTRATION

Richard S. Leblang
National Weather Service Forecast Office
Bismarck, North Dakota

1. Introduction

Electronic computer implementations of artificial intelligence fall under two categories. In an "expert system" the programmer encodes a series of inviolable rules which the computer follows in making a decision. This approach requires a complete understanding of the problem to be solved. The computer is incapable of original thinking and unforeseen input patterns may cause the program to fail. Another approach to artificial intelligence, neural networks, can overcome these deficiencies when properly implemented.

Neural networks (Jones and Hoskins, 1987; Stanley, 1988; Touretzky and Pomerleau, 1989) seek to emulate the structure and functioning of the human brain on the neuron level. The basic element of a neural network is the processing unit (Fig. 1). A processing unit usually receives input from several other processing units. These inputs are summed by a designated summation function (most often a simple arithmetic sum is employed). This sum is then put through a threshold function. Viewed simply, if the sum exceeds a certain predetermined value, then the threshold function allows a non-zero value to be sent out as output to other processing units further down the line.

The links between the processing units are the heart of the neural network, for it is here that the learned "knowledge" of the network resides. Each link has a weighting value (usually between 0 and 1) that is uniquely its own. When the output of one processing unit is sent along a link to become the input of another processing unit, it is first multiplied by the weighting value of that particular link. During the network learning process, these weighting values are adjusted until the network has reached the required level of intelligence. Figure 2 demonstrates how actual values flush through a portion of a neural network. The input values of .5 and 1 are each multiplied by the weighting values associated with the inbound links to the processing unit. The results of these multiplications are summed at the processing unit and if they exceed the threshold value of that unit (which they do in this case) the sum is then sent out to the next layer of processing units.

Best
Available
Copy

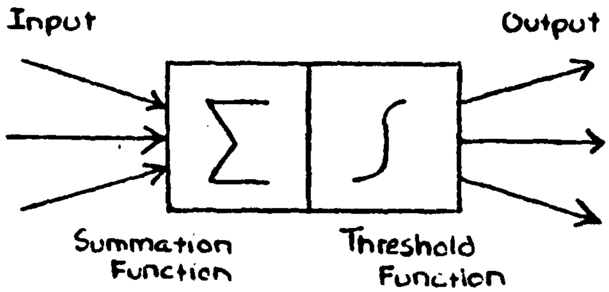

Figure 1. Basic processing unit.

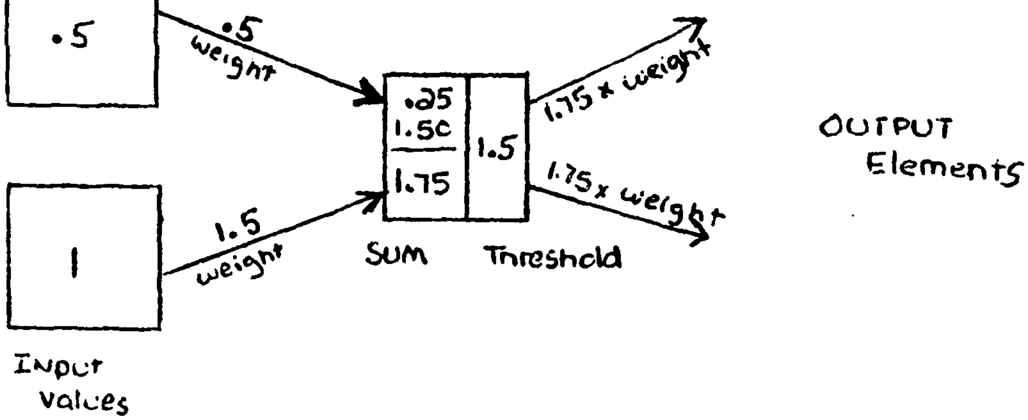

Figure 2. Typical unit calculation.

There are several different algorithms that can be used to train neural networks to handle a specific task. One of the most popular is the back-propogation rule (Jones and Hoskins, 1987). In back-propogation the network is shown a series of training pairs; each consisting of an input pattern and the expected final output pattern. The weighting values are initially randomly set and the first input pattern is flushed through the network. The values at the last (or output) layer of processing units is then compared to the expected answer and differences at each output unit are determined. These differences are then used to calculate small corrections to the weighting values, back through the network to the original input layer of processing units. The procedure is repeated for all the other training pairs several times over and over until the level of error falls below a predetermined value. Thus, if you have 100 training pairs and need to cycle through them 200 times to reach a level of acceptable training you would need to execute 20,000 training iterations. Clearly, training a complex network is computationally intensive; however, once it is trained, it can solve problems very fast, as only one pass through the system is required.

The design of most neural networks include at least one or more "hidden" layers (Touretzky and Pomerleau, 1989; Stanley, 1988) between the input and output layers. It has been shown that a minimum of three layers is required to duplicate most basic logical operations. Figure 3 illustrates a hypothetical three layer network with its inter-connections (links).

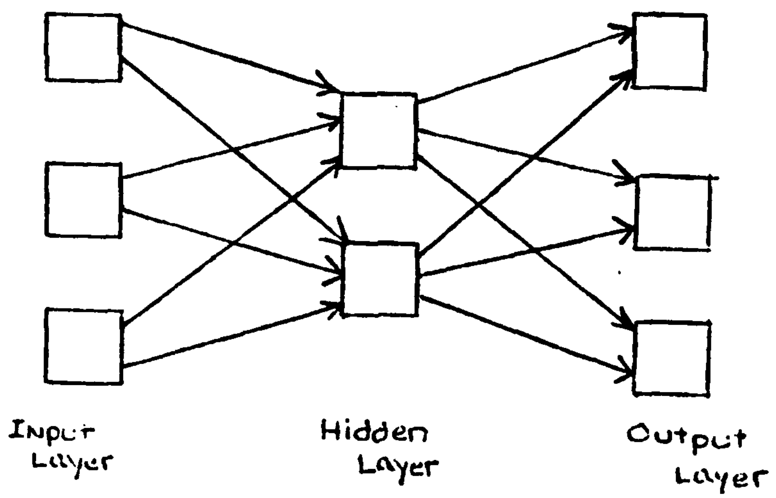

Figure 3. Hypothetical 3 layer neural network.

Neural networks excel in pattern recognition. Once trained, they are over twice as fast as an expert based system at this task. They also have the ability to make reasonable guesses when faced with new patterns on which they were not trained. This ability to handle "fuzzy logic" is lacking in expert based systems.

Because many rules and techniques in weather forecasting employ pattern recognition of one sort or another, the author believes that neural networks hold much promise as a forecasting tool. To demonstrate this, a simple neural network was designed to forecast a 12 hour 500 mb height tendency at a single point.

2. The Experiment

A neural network was designed and trained to forecast a 12 hour 500 mb height rise, fall, or no change at a single point. Input to the network were gridded 500 mb heights over North America.

To this end, 116 plot files containing 500 mb height data from the fall of 1989 through the spring of 1990 were analyzed and gridded to 13 x 10 point grid fields using a Cressman type biquadratic interpolation. Grid Point (7,5) was selected as the forecast point and the actual 12 hour height tendency (rise, fall, no change) was determined from the observed data. To further simplify the problem, the mean height was determined for each grid field, and then each grid point was assigned a value of 1 or 0 depending upon whether its height value was above or below the mean field height. Figures 4 and 5 illustrate a height analysis and how it was transformed for input into the neural network.

The neural network itself consisted of three layers. The input layer consisted of 130 processing units, each one corresponding to a grid point on the grid field. The second layer consisted of 11 hidden units. The third, or output layer, consisted of three elements, one representing a height rise, another no-change, and the last a height fall. Once trained, the network is presented with the 130 values (0's and 1's) of a height field. These values are flushed through the system to the output layer where the processing unit (rise, fall, or no change) with the highest value represents the network's 12 hour height tendency forecast at grid point (7,5).

To train the network, 94 of the height fields along with their observed 12 hour height tendencies at grid point (7,5) were selected. The neural network, which was implemented in QuickBasic 4.00 and employed a back-propogation learning algorithm, required over two hours to run through the 94 training pairs 300 times. An IBM clone with a 80286 processor running at 20 mhz along with a math coprocessor was used for the computations. The remaining 22 input fields were then used for test forecasts to see how well the network performed.

3. The Results

After the network was trained and its weighting values permanently set, the original 94 training input fields were run through the network. In every instance the correct height tendency was returned. Thus, the network was able to learn and recognize all of the 94 input patterns. The remaining 22 input fields were then run through the network and its forecast for each one was compared to the actual height tendency. The network correctly forecast the tendency for 15 of the 22 test cases (68 percent accuracy). Two of the test cases involved a no-change situation. The network failed to predict a no-change value for both of these cases. Considering that only one height change (0 meters) will result in this outcome; whereas a whole range of height changes encompass a rise or

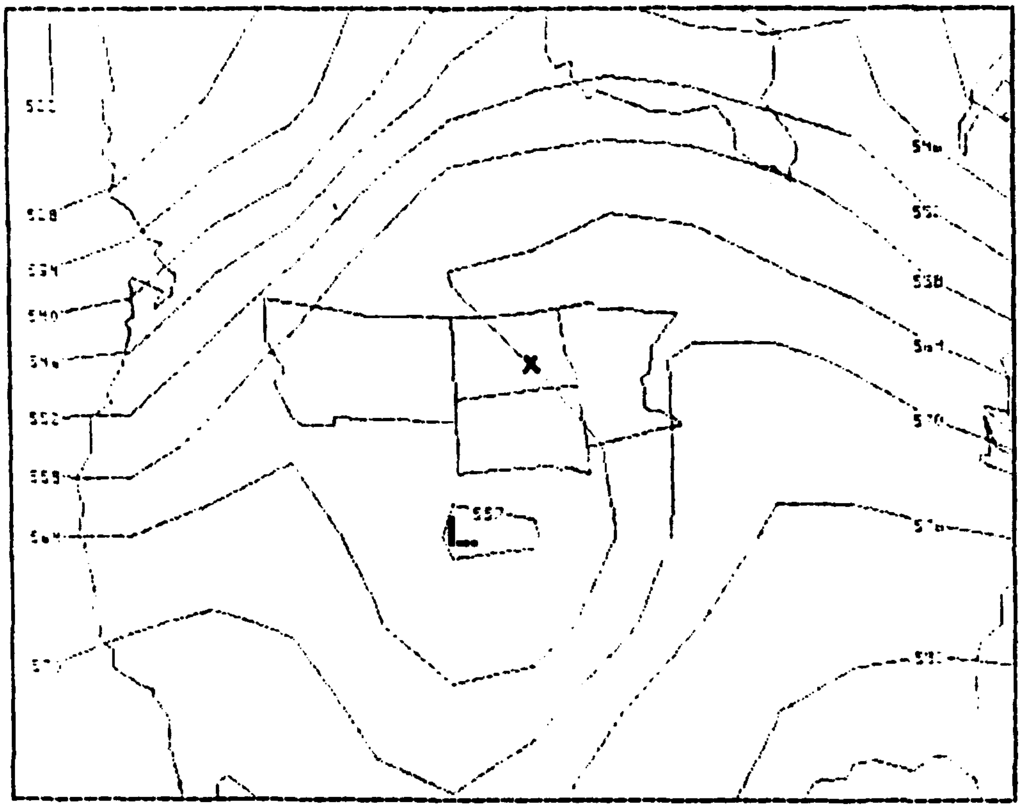

Figure 4. 500 mb height analysis, March 8, 1990 at 00Z.

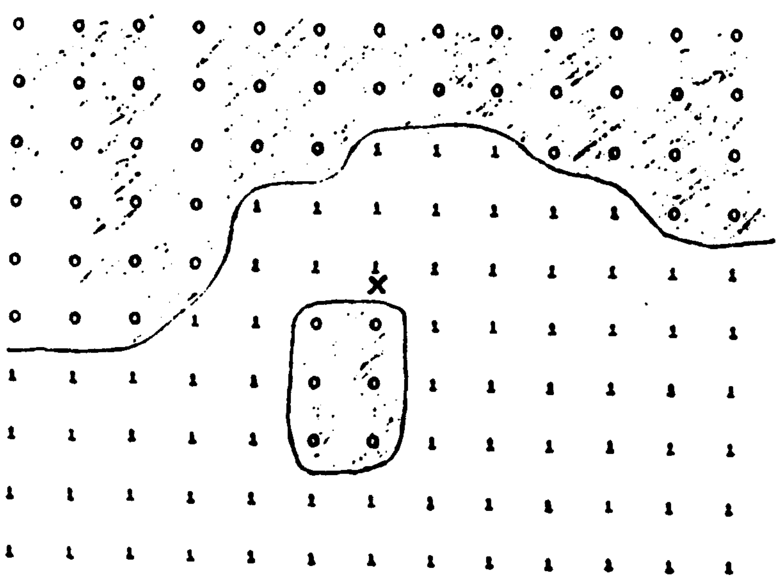

Figure 5. Neural network binary input field, March 8, 1990 at 00Z. "X" marks location of the forecast point. Shaded areas of the "O's" are low pressure.

fall, this result is not surprising. If we ignore the no-change cases, the overall forecast accuracy of the network improves to 15 out of 20 or 75 percent.

4. Interpretation of the Hidden Layer Units

The 11 processing units in the hidden layer act as feature detectors. Each hidden unit is excited by a particular input pattern or combination of input patterns. There are three links from each hidden unit to each output unit. By examining the weighting value for each of these links it can be determined whether a particular hidden unit is a feature detector for rising, falling, or steady heights. For example, the weighting value associated with the link connecting hidden unit 3 with output unit 1 (the output unit associated with rising heights) was -1.9 after the network training phase. Likewise, the weighting value associated with the link to output unit 2 (the output unit assigned steady heights) was -1.7 after training. But, the weighting value along the link between hidden unit 3 and output unit 3 (the one associated with falling heights) was +3.4 after training. Thus, hidden unit 3 is a feature detector for height patterns related to falling heights at the forecast point. In other words, when the 130 input values of a height pattern associated with a falling height at the forecast point are presented to the network, the sum of these input values (0's and 1's) times the unique weighting values connecting each input unit with hidden unit 3, will exceed the threshold value of hidden unit 3. It then in turn sends a value (nearly 1.0) along its three links to the output units. Only output unit 3 (falling heights) will receive a positive value. The hidden unit has thus cast a strong vote for falling heights. Only after the votes from all 11 hidden units are counted is the final forecast determined.

It can be instructive to graphically display the weighting values from the 130 input units to selected hidden units representative of rising and falling heights. That way one can see what the network has learned about interpreting height maps. Remember that in a neural network, the programmer does not imbue the program with predetermined knowledge or rules, these are learned by the network during the training phase when it is presented with the input examples and the desired output. What the network has learned on its own can be helpful to the forecaster in his own interpretation of height patterns.

Figure 6 is a graphical representation of the weighting values along the 130 input links leading to hidden unit 4. To reduce the size of the representation, each weighting value displayed is actually a regional sum of the four nearest weighting values. The rightmost column of original weighting values was not used in this analysis. The location of the forecast point is indicated by the "x". Areas of negative numbers have been shaded and are representative of where low heights are needed to prevent the deactivation of this hidden unit. The positive numbers indicate where high heights should be located for this hidden unit's activation. Hidden unit 4 is a feature detector for falling heights at the forecast point. This is immediately apparent because its activation requires a broad area of low heights just upstream from the forecast point.

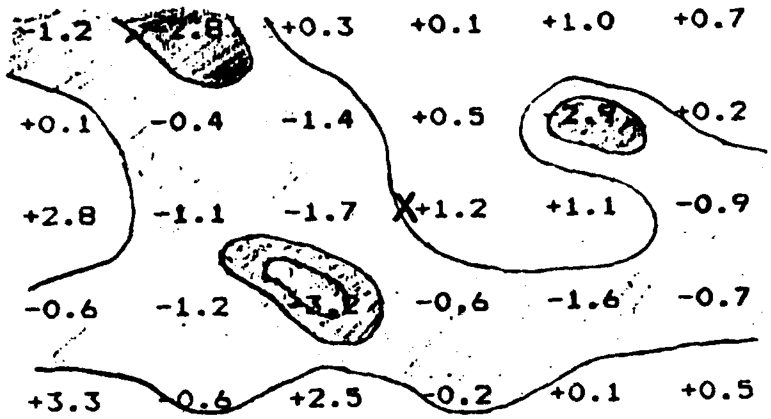

Figure 6. Activation map of hidden unit 4, a feature detector of falling heights at point "X". Shaded negative numbers are areas of lower pressure. "X" marks the forecast point.

Figure 7 is a representation of hidden unit 3. It also is a detector of falling heights at the forecast point. Because it has a narrow band of higher heights just upstream of the forecast point, its meaning is perhaps a little more obscure. It just so happens that the height pattern displayed in Figures 4 and 5 strongly activated this hidden unit; and indeed the actual 12 hour height change at the forecast point was negative. Examination of the pattern indicates a closed low south of the forecast point and a long wave trough to the far west. An area of weak relatively higher heights lay between these two features. The graphical representation of hidden unit 3 does mimic this input pattern to a certain extent.

Figure 8 shows the activation pattern of hidden unit 6 which is a strong detector of rising heights at the forecast point. This pattern is easy to fathom with is strong area of higher heights just upstream of the forecast point and a deep lower height area just to the east.

Some of the other activation patterns of the remaining hidden units are not so straightforward and this is what makes neural networks so fascinating. Has the network learned something about height pattern recognition which we have so far failed to grasp?

5. Conclusions

This experiment demonstrates the efficacy of using neural networks to solve certain forecast problems. There is a presumption, of course, that similar weather patterns will lead to similar weather events in time. This, as we all know, is not always the case. But, as we become more detailed in describing a pattern (i.e., using more and more meteorological parameters) we can only improve this approach. To analyze and recognize these complex patterns we will need to rely more and more on neural networks and their advanced pattern recognition capabilities. Imagine a neural network whose input is the actual 500 mb heights and whose output is the 12-48 hour forecast of heights at each grid point. Also, imagine that its training pairs include all the upper air data

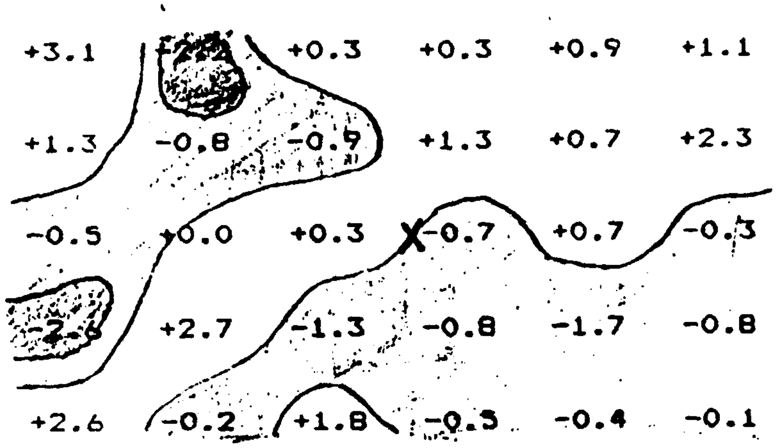

Figure 7. Activation map for hidden unit 3, a feature detector of falling heights at point "X".

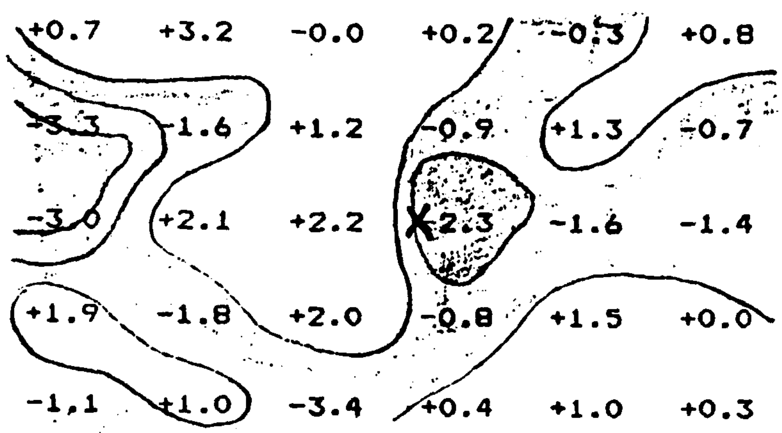

Figure 8. Activation map for hidden unit 6, a feature detector of rising heights.

from the last 30 years. Such a network would have to be trained on a super-computer. But lets say that after its training is complete, it can forecast 500 mb heights with 75 to 85 percent accuracy in a tiny fraction of the time it takes to run a complex physical model. Then perhaps we will have gained a powerful tool that will complement our detailed forecast models.

6. References

Jones, W. P., and J. Hoskins, 1987: Back-Propogation, Byte, 12, 155-162.

Stanley, J., 1988: Introduction to Neural Networks, California Scientific Software, Sierra Madre, CA, 255 pp.

Touretzky, D. S. and D. A. Pomerleau, 1989: What's Hidden in the Hidden Layers? Byte, 14, 227-233.