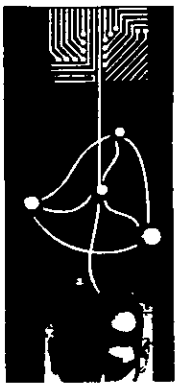

1° Congresso Brasileiro de Redes Neurais

Escola Federal de Engenharia de Itajubá
Itajubá, 24 a 27 de outubro de 1994

STRUCTURING NETWORKS FOR IMAGE CLASSIFICATION USING COMPETITIVE LEARNING

P. R. Green[†], C. L. Nascimento Jr.[‡] and T. A. York[†]
[†] UMIST, UK and [‡] Instituto Tecnológico de Aeronáutica, Brazil
E-mail: p.r.green@umist.ac.uk, cairo@ieav.cta.br, tay@umist.ac.uk

Abstract: One major problem facing the designer of a multilayer feedforward neural network for image classification is to determine the optimum number, and size, of the hidden layers. In this article we propose to add competition within the hidden layer to encourage structuring of the solution during Back-Propagation training. Simulation of an image classification network with added competition shows that a subset of hidden layer units account for most of the network solution. As a result, the remaining hidden layer units can be removed without causing significant performance degradation.

1. INTRODUCTION

One major problem facing the designer of a multilayer feedforward neural network for image classification is to determine the optimum number, and size, of the hidden layers. A popular approach is to initially design with one hidden layer, guess the number of units and use the Back-Propagation algorithm to train the network through adaptation of the weights, Hinton (1).

The Back-Propagation algorithm minimises the average squared output error over the set of training patterns without regard to the structuring of the solution in the network. Consequently, given an overestimated number of hidden layer units, Back-Propagation will distribute the solution over the whole hidden layer, contrasting with a structured solution which uses only the required minimum of units. The unstructured solution, requiring all hidden units, has clear associated implementation penalties and the potential for degraded classification of noisy images.

In this article we propose to add competition within the hidden layer to encourage structuring of the solution during Back-Propagation training. Simulation of an image classification network with added competition shows that a subset of hidden layer units account for most of the network solution. As a result, the remaining hidden layer units can be removed without causing significant performance degradation.

2. BACKGROUND

Multilayer feedforward neural networks have been successfully applied to a diverse range of data classification and recognition problems. Of these, image classification is a current focus as few 'traditional' computing solutions offer the robustness the problem demands, Zurada (2). In these networks, the hidden layer adapts to form a set of feature detectors that are then used by the output layer to classify the image. The adaptation of the hidden layer in developing these feature detectors is one of the major innovative components of the neural network approach.

In contrast to feedforward networks trained with supervised learning, such as the Back-Propagation algorithm, other neural network models use competition amongst the units of a layer and unsupervised learning to discover a set of feature detectors, Rumelhart and Zipser (3). The mutual inhibition between units of a layer in competitive learning encourages the partitioning of the input pattern space into features for which a minimum number of units are active. This unsupervised

classification, also termed *clustering*, is the basis of a number of neural models such as the Kohonen network, Kohonen (4), and the Adaptive Resonance Theory 1 (ART1) network, Carpenter and Grossberg (5).

The addition of competition transforms a feedforward network to a feedback network. Back-Propagation was originally conceived for feedforward networks (1), and has subsequently been extended to certain classes of feedback networks, Rumelhart et al (6) and Almeida (7). However, the network model presented in this report retains the Back-Propagation algorithm for feedforward networks.

3. BACK-PROPAGATION WITH ADDED HIDDEN LAYER COMPETITION

The proposed network model, shown in figure 1, consists of input, hidden and output layers, as found in the standard multilayer feedforward neural network, but differs in that the units of the hidden layer are also linked by competitive weights. Figure 1 shows the weights for a typical unit in each layer (I_q, H_p, O_m). The bias weights for the hidden and output layer units are omitted for clarity.

Feedforward Behaviour

Given that; NI, NH and NO are the number of input, hidden and output units respectively, $\boldsymbol{I}$ is the external input vector applied to the network and $\boldsymbol{H}$ and $\boldsymbol{O}$ are the outputs of the hidden and output units, then the vector $\boldsymbol{O}$ is calculated as follows:

$$\boldsymbol{I} = \left[I_1, ..., I_q, ..., I_{NI} \right]^T$$

$$\boldsymbol{H}^k = \left[H_1^k, ..., H_p^k, ..., H_{NH}^k \right]^T$$

$$\boldsymbol{O} = \left[O_1, ..., O_m, ..., O_{NO} \right]^T$$

$$\boldsymbol{net}_H = \boldsymbol{W}_{HI} \boldsymbol{I} + \boldsymbol{bias}_H \qquad (1)$$

$$\boldsymbol{H}^k = Sig\left(\boldsymbol{W}_{HH} \boldsymbol{H}^{k-1} + \boldsymbol{net}_H \right) \qquad (2)$$

$$\boldsymbol{O} = Sig\left(\boldsymbol{W}_{OH} \boldsymbol{H}^{NT} + \boldsymbol{bias}_O \right) \qquad (3)$$

where in the recursive eq. 2, k = 1, ..., NT, $\boldsymbol{H}^0$ is a zero vector, $Sig(X)$ is the sigmoid function, $[1+\exp(-x)]^{-1}$, applied to each component of the vector X, and the input vector $\boldsymbol{I}$ is kept constant during the recursion. The weight matrices $\boldsymbol{W}_{HI}$ and $\boldsymbol{W}_{OH}$ have dimension NH by NI and NO by NH respectively. The column bias vectors $\boldsymbol{bias}_H$ and $\boldsymbol{bias}_O$ have dimensions NH and NO. The competition matrix $\boldsymbol{W}_{HH}$, dimension NH by NH, is given by:

$$W_{HH}^{ij} = \begin{cases} \gamma < 0 & if\ i \neq j \\ 0 & if\ i = j \end{cases} \qquad (4)$$

and γ is called the *competition factor*. Note that if the competition factor is set to zero, the network model reduces to the standard feedforward multilayer network.

Network Training

Given a set of NPAT training images and their desired classifications (T_{pat}, pat = 1, ..., NPAT), the network is trained using standard Back-Propagation. A training epoch consists of the presentation of all NPAT training images. For each training image I_{pat} presented, the outputs of the hidden layer H_{pat}^{NT} and the output layer O_{pat} are calculated according to eqs. 1, 2 and 3. The error signal, defined as the difference between the desired and actual network outputs for a given image, is then back-propagated through the network.

Training adapts the matrices $\boldsymbol{W}_{HI}$ and $\boldsymbol{W}_{OH}$ and the vectors $\boldsymbol{bias}_H$ and $\boldsymbol{bias}_O$ following the presentation of each image. The error signal is not allowed to back-propagate through the competitive weights in the hidden layer. The matrix $\boldsymbol{W}_{HH}$ is not adapted.

Following each epoch, we can form a measure of the classification error by calculating the Root-Mean-Square (RMS) Error over the set of NPAT training images as:

$$RMS\ Error = \frac{1}{NO} \sqrt{\sum_{pat=1}^{NPAT} E_{pat}^2} \qquad (5)$$

where:

$$E_{pat}^2 = \left(T_{pat} - O_{pat} \right)^T \left(T_{pat} - O_{pat} \right) \qquad (6)$$

4. AN IMAGE CLASSIFICATION PROBLEM

A series of simulations were designed around a simple image classification problem for the purpose of demonstrating the learning dynamics and knowledge structuring of the proposed network model.

The problem required sixteen unique bitmapped images to be classified using a 1-of-16 line coding scheme. The images used were taken from the 8 by 8 pixel character bitmaps used by the IBM PC XT, IBM (8). These were edited to remove the redundant last column and row to form the 7 by 7 bitmaps shown in figure 2. The sixteen characters images were drawn from the uppercase hexadecimal

character set (0,1-9,A-F).

The simulated network had an input layer with 49 units, a single hidden layer with 10 units and an output layer with 16 units. Each input unit was connected to a single pixel of the image bitmap with active pixels having an input value of 1 and inactive pixels 0. The output layer implemented the 1-of-16 line classification coding.

5. SIMULATIONS

The simulations consider two particular cases:

I. No competition, $\gamma = 0$
II. With competition, $\gamma = -1.4$

The network weights and biases, excluding the competitive weights, were initialized as small random numbers with a uniform distribution around 0. Excluding the competitive weights, in both cases the network initial weights and biases are exactly the same. The learning rate and momentum rate were set to 0.5 and 0.1 respectively. For the case with competition, the number of recursions through the hidden layer (NT) was set to 2.

Figure 3 shows the learning curve for the cases with and without competition for 1200 epochs. The RMS Error was calculated every 4 epochs. During each epoch, all 16 input patterns were presented with a different random ordering and without noise. The ordering of the presentations of the patterns is the same for both simulations.

In both cases the network learned to classify the input images. Figure 4 shows, for the case without competition, the output of the hidden layer at the end of training for all 16 training input patterns. Figures 5 and 6 show, for the case with competition ($\gamma = -1.4$), the output of the hidden layer following the first recursion (H^1) and following the second recursion (H^2).

6. DISCUSSION

Figure 4 shows that, for the case without competition, each hidden unit is activated by several images. Figure 5 shows that, when the competition was added, after the first recursion through the hidden layer (k=1), some hidden units, in particular units 2 and 10, have small outputs for all training inputs. Figure 6 shows that after the second recursion (k=2), hidden units 1,2,3,7 and 10 have zero output values.

Considering figure 6, it could be easily assumed that hidden units 1,2,3,7 and 10 could be removed from the network without any degradation in classification. This is not the case because the network is not simply feedforward. A near zero output from a hidden unit at the end of recursion does not imply near zero output on earlier iterations. Consequently, such a unit could have played an active role in the competition mechanism and can not therefore be removed.

From eq. 2, we can see that, due to the competition within the hidden layer and the fact that all hidden units use a sigmoid function, during recursion the output of a particular hidden unit will always be smaller or equal to its value after the first recursion cycle, ie, $H_i^k \leq H_i^1$.

From the above discussion, it can be seen that if the output of a particular hidden unit is zero after the first recursion cycle for all input training images then, such a hidden unit can be removed without any degradation in classification. Hence, the removal of a hidden unit will result in a small degradation if it has small output for all input training images after the first recursion.

This suggests that hidden units can be progressively removed in exchange for a measured degradation in classification. One possible criterion for selecting hidden units would involve a simple threshold function. Defining $H_i^{1\,MAX}$ as the maximum output, over all input patterns, from hidden unit i ($i = 1, \ldots, NH$) after the first recursion. Hidden unit i would be removed if:

$$H_i^{1\,MAX} < \theta \qquad (7)$$

where θ is a threshold level selected by the designer.

Tables 1 and 2 show the effect of selecting different threshold levels for the simulated networks trained with and without competition. The *relative RMS error* is defined as the ratio between the RMS error when a particular set of hidden unit (HU) is removed and the RMS error with no HUs removed. We define a classification as valid when a single unit in the output layer has an output: 1) greater than half of the summation of all output layer units and 2) greater than 0.5.

For the case without competition, H_i^{MAX} is near 1 for all hidden units. This indicates the need for large values of threshold and also that no hidden units can be removed without a significant performance.

TABLE 1 - Threshold for case without competition
(RMS Error no HUs removed = 0.0134)

Threshold θ	HUs Removed	Relative RMS Error	Misclass-ifications
0.9000	none	1.0	0
0.9971	3	4.7	1
0.9980	1,3	8.5	4
0.9981	1,3,5	10.7	8
0.9983	1,3,5,7	15.8	13
0.9990	1,3,5,7,9	19.4	14

TABLE 2 - Threshold for case with competition
(RMS Error no HUs removed = 0.0526)

Threshold θ	HUs Removed	Relative RMS Error	Misclass-ifications
0.10	none	1.00	0
0.40	2	1.02	0
0.60	2,10	1.17	0
0.70	1,2,10	1.55	2
0.89	1,2,7,10	2.21	2
0.90	1,2,3,7,10	2.72	3

TABLE 3 - Single HU failure without competition

Hidden Unit	RMS Error	Relative RMS Error	Misclass-ifications
-	0.013	1.0	0
1	0.050	3.7	0
2	0.116	8.7	4
3	0.063	4.7	1
4	0.105	7.9	5
5	0.071	5.3	2
6	0.112	8.3	5
7	0.079	5.9	1
8	0.100	7.4	2
9	0.070	5.2	2
10	0.116	8.6	4

TABLE 4 - Single HU failure with competition

Hidden Unit	RMS Error	Relative RMS Error	Misclass-ifications
-	0.053	1.0	0
1	0.058	1.1	0
2	0.054	1.0	0
3	0.063	1.2	0
4	0.106	2.0	2
5	0.216	4.2	7
6	0.219	4.2	8
7	0.064	1.2	0
8	0.213	4.0	8
9	0.213	4.0	7
10	0.056	1.1	0

degradation, as shown by the relative RMS error column.

In contrast, in the case with competition, there are units with relatively small $H_i^{1\ MAX}$ which can be removed without significant performance degradation.

One incorrect inference of the results given in table 2 would be that, training with competition has increased the tolerance of the network to the loss of *any* small group of hidden units. Consider the results of tables 3 and 4 where the effects of removing a single hidden unit are shown for networks trained with and without competition. The relative RMS error column for the case with competition reveals two distinct classes of hidden unit. Hidden units 1,2,3,7 and 10 form one class, where each could be removed with less than a 20% increase in the RMS error. Removal of any one unit in the other class, consisting of units 5,6,8 and 9, would cause a significant loss of classification performance.

The existence of distinct classes clearly shows that training with competition encourages structuring of the solution and not an increased tolerance of the network to the loss of *any* hidden layer unit.

For the network trained without competition, table 4 shows no particular divisions in the relative RMS error of the hidden units and hence no structuring.

CONCLUSIONS AND FUTURE WORK

This article has presented a neural network model which uses competition in the hidden layer of a feedforward network to encourage structuring during Back-Propagation supervised learning. The result of this structuring is to reduce, over the feedforward network, the number of hidden layer units active in feature detection. Compact structuring and fault-tolerance are seen to be conflicting requirements. These issues were demonstrated with the simulation of a simple image classification problem. The benefits offered by this new network model must be weighed against the need to tune an additional parameter, the competition factor. Simulations results not presented here indicate that too larger competition can inhibit learning.

There are a number of aspects of the new model that merit further work, including; modification of the learning algorithm to remove 'inactive' hidden layers units during training, scheduling the competition factor γ to improve the time taken to train the network to a given accuracy of classification, investigation of the effects of the number of cycles through the hidden layer (NT) on training and classification performance.

ACKNOWLEDGMENTS

The authors would like to acknowledge Dr. M. Zarrop for his interest in this work.

Cairo L. Nascimento Jr. gratefully acknowledges the support of the Brazilian Research Council (CNPq) under grant 200.617/88.5.

REFERENCES

1. Hinton, G.E., 1989, Artificial Intelligence, 40, 185-234

2. Zurada, J.M., 1992, "Introduction to Artificial Neural Systems", West Publishing Co.

3. Rumelhart, D.E. and Zipser, D., *Feature Discovery by Competitive Learning*. In Rumelhart, D.E. and McClelland, J.L. (eds.), 1986, "Parallel Distributed Processing", Vol. I, MIT Press, 151-193

4. Kohonen, T., March 1988, IEEE Computer Magazine, 11-22

5. Carpenter, G. and Grossberg, S., 1986, "Eighth Annual Conference of the Cognitive Science Society", 45-62

6. Rumelhart, D.E., Hinton, G.E. and Williams, R.J., *Learning Internal Representations by Error Propagation*. In Rumelhart, D.E. and McClelland, J.L. (eds.), 1986, "Parallel Distributed Processing", Vol. I, MIT Press, 318-362

7. Almeida, L.B., *Backpropagation in Non-Feedforward Networks*. In Aleksander, I., 1989, "Neural Computing Architectures", North Oxford Academic, 74-91

8. IBM, 1986, "Technical Reference: Personal Computer XT and Portable Personal Computer"

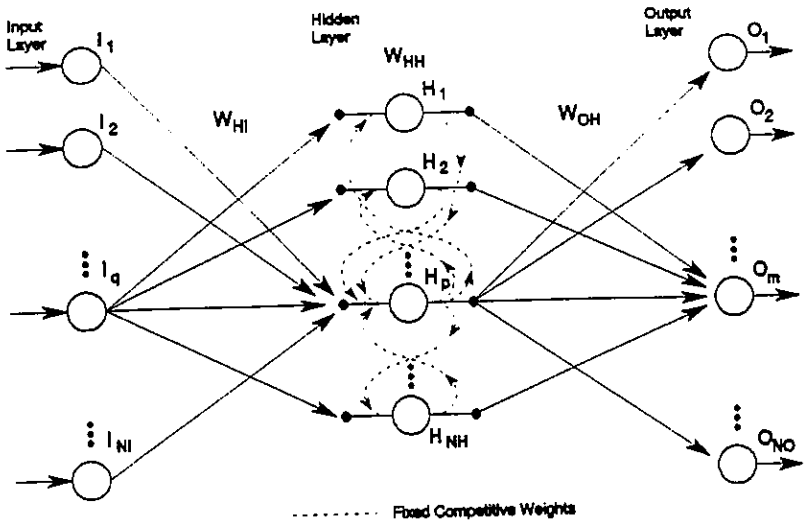

FIGURE 1 - Proposed Neural Network Model

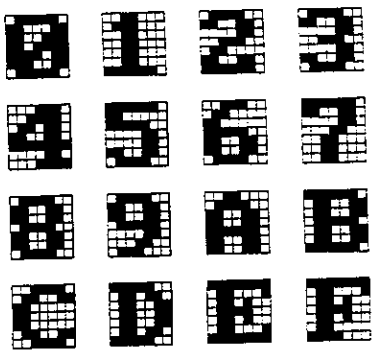

FIGURE 2 - Image bitmaps

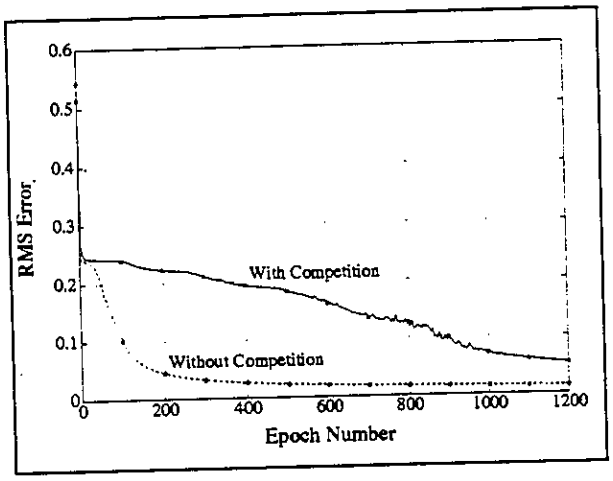

FIGURE 3 - Learning Curves

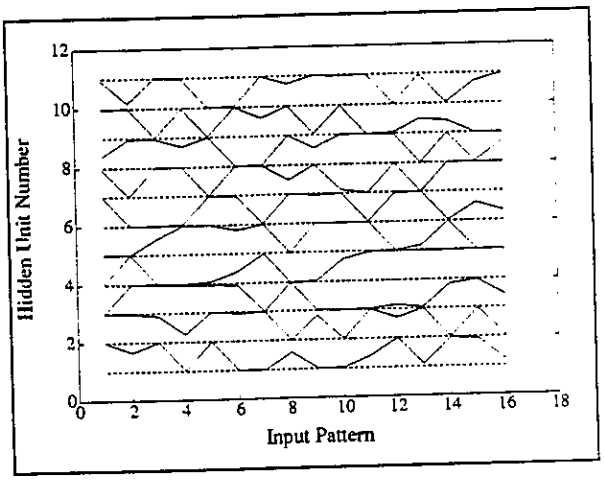

FIGURE 4 - Hidden Unit Outputs - No Competition

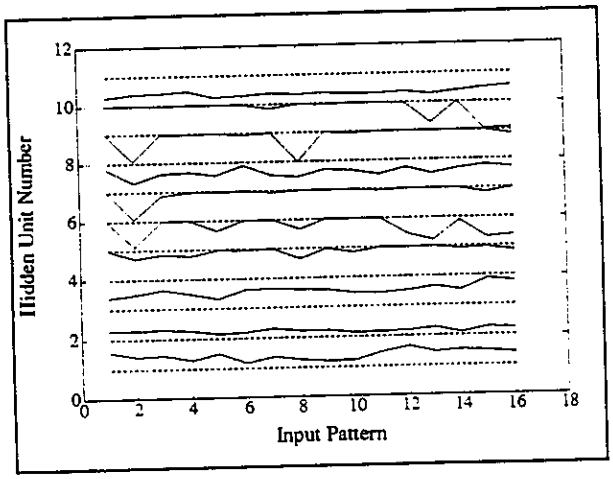

FIGURE 5 - Hidden Unit Outputs - With Competition First Recursion

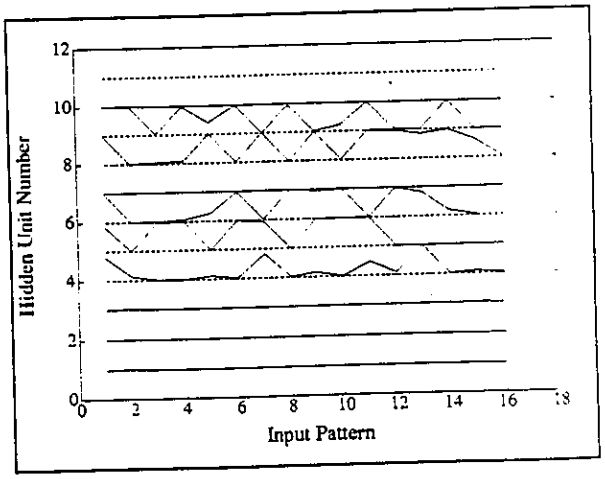

FIGURE 6 - Hidden Unit Outputs - With Competition Second Recursion