Neural Network Applications

E. Vonk *, L.C. Jain **, and L.P.J. Veelenturf *

*Control, Systems and Computer Engineering Group (BSC),
Laboratory for Network Theory,
Department of Electrical Engineering,
University of Twente, Postbus 217,
7500 AE Enschede,
The Netherlands.
Tel. : +61 8 302 3984
Fax : +61 8 302 3384
Email: 940021a@lux.levels.unisa.edu.au

**Knowledge-based Engineering Systems Group,
School of Electronic Engineering,
University of South Australia,
Adelaide, The Levels, 5095,
Australia.
Tel. : +61 8 302 3315
Fax : +61 8 302 3384
Email: etlcj@lv.levels.unisa.edu.au

Keywords neural network, Back Propagation, Radial Basis Function, Kohonen, ART1, Hopfield, Bidirectional Associative Memory, character recognition

Abstract

Artificial neural networks, also called neural networks, have been used successfully in many fields including engineering, science and business. This paper presents the implementation of several neural network simulators and their applications in character recognition and other engineering areas.

1. Introduction

In the last five years the reported applications of neural networks have multiplied exponentially. These networks have proven to be successfull in many applications including fault diagnosis, financial control, forecasting, plant control, industrial process control and pattern recognition.
Despite tremendous interest in neural networks, there are still many research problems which need careful attention. For example, in the back propagation network, parameters such as learning rate and momentum term are found by trial and error.
The mathematical basis for finding the number of neurons in the hidden layer are reported in the literature [2], but there are still some problems associated with it. Often the researchers adjust the neurons in the hidden layer by trial and error. There are not enough guidelines pertaining to which architecture is a suitable for a given application.

This paper presents the implementation of back propagation, radial basis function, Kohonen net, ART1, Hopfield net and BAM. Other applications are also presented.

2. Neural network simulations

This section describes the neural network simulators and their respective software implementations used in this study.

2.1: The Back propagation network (BP).

The back propagation network consists of an input layer of neurons (not shown), an output layer of neurons and at least one hidden layer as shown in Figure 1.

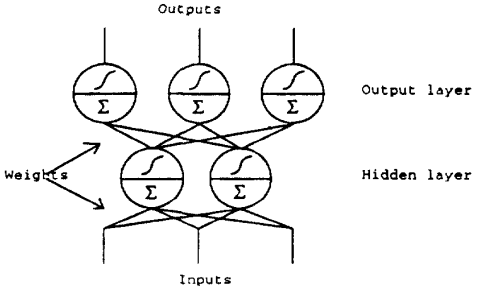

Fig. 1:A '3-2-3' Back Propagation Network.

The neurons perform a weighted sum of their inputs and use this sum as the input of an activation function which is usually the sigmoid function. A supervised learning algorithm [2] [8] is used to teach the network. It consists of updating all the weights of a network untill the output of the network reaches the pre-specified desired output. The factors responsible for the training and performance of the network include:

- the initial values of the weights (usually random)
- the number of training cycles
- the number of hidden neurons
- the training set
- the values of the teaching parameters (learning rate, momentum term)

0-8186-7085-1/95 $04.00 © 1995 IEEE

2.2: The Radial basis function (RBF) network [4] [6] [7].

The Radial Basis Function network consists of three layers. The hidden layer is used to cluster inputs of the network; neurons in this layer are therefore called cluster centres. Though its architecture is similar to a three layer back propagation network, its working is different. It uses a Gaussian kernel function to calculate the activations of the neurons in the first layer. The neurons in the output layer perform an ordinary linear weighted sum of these activations. The learning in this network is done in two stages. Firstly, the input-training patterns are clustered (unsupervised) to their nearest cluster centres by means of a clustering algorithm. The K-Means clustering algorithm is used. After this, the spread of each cluster centre is determined. This can be done in several ways; they can be made equal to the average distance between the cluster centre and the training patterns clustered with that centre [4], or they can be set equal to the average distance between the cluster centre and its nearest cluster centres (often two) [6].

The second stage of learning is the supervised learning of the weights of the output layer. This is usually done by the least mean square algorithm. The inital value of the output weights are usually set to small random values.

The performance of the Radial Basis Function network depends on the way the inputs are clustered. As with the back propagation network, the number of neurons in the first layer greatly influences the performance of the network. The training time of this network is typically orders of magnitude smaller than that for the back propagation network as training is split up in two parts, both of which can be done quite fast. In our program the only teaching parameter is the learning rate. The calculation of training and test errors are identical to the back propagation program.

2.3: The Kohonen self organising feature map [5].

The Kohonen self organising feature map has two layers of neurons. The signals from input neurons (also called inputs) are fed to every single neuron. While learning (unsupervised) the network makes a two dimensional representation of the input space. The multi-dimensional input space is transferred to a two-dimensional grid using this network. Fig. 2 shows a Kohonen network that has two inputs and a grid (or feature map) consisting of 3 by 3 = 9 output neurons.

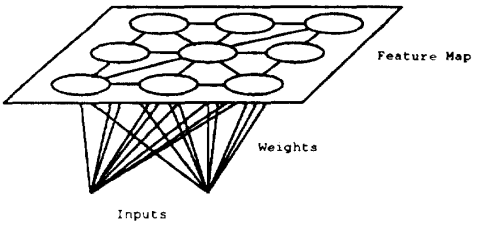

Fig. 2.: A '2-3-3' Kohonen Network.

Learning in the Kohonen network is performed by selecting the winning neuron for a certain training input. This neuron has the weights that most closely resemble the values of the inputs. The weights of this neuron as well as of its neighbours are updated to resemble like the input vector. The neighbours are determined by a neighbourhood function. The radius of this neighbourhood function determines how many neurons around the winning neuron are included in the neighbourhood. There are several shapes possible for the neighbourhood function. A square shape, a bubble (circle) shape or a hexagonal shape are the ones usually used.

In order to train the network properly both the learning rate and the radius of the neighbourhood function should decrease in time. The initial weights of the network should be different from each other and are often set to random values in the range of the possible input values.

The learning process consists of two phases. In the first phase, coarse learning, the learning rate is quite large and the neighbourhood radius decreases from encompassing a large part of the grid to encompassing only the winning neuron itself (radius = 0) or the winning neuron and its closest neighbours (radius = 1). In the second phase the learning rate slowly decreases to zero (this is usually the stopping criterion) while the neighbourhood radius remains 0 or 1.

The learning rate as well as the neighbourhood radius can decrease by a number of functions; e.g. linearly, exponential. According to [5] this is not very critical. The total number of training steps (= the number of times a single input training pattern is presented to the network) is typically 500 times the number of network units or less, with the first phase of learning lasting only about 1000 cycles.

2.4: The ART1 clustering algorithm [8]

The ART1 network consists of two layers; a layer to which the inputs are applied and a top layer of neurons (outputs). The two layers are fully

interconnected with feedforward (from input-layer to top-layer) weights (the 'W-weights') as well as feedback weights (the 'V-weights').

The 'W-weights', or clustering weights, are used to select the winning output neuron (the cluster) and serve as the long term memory of the network. The winning neuron is the neuron with the maximum response (= weighted sum) for a certain input pattern. The 'V or vigilance weights' are used for the vigilance test and serve as the short term memory of the network. When a training input is similar enough to the last pattern clustered with the winning neuron it will be clustered with this neuron. Otherwise the neuron with the second largest response is tried. The required similarity is determined by the vigilance threshold. If the input does not cluster with any of the existing clusters (output neurons), then a new cluster centre is made. The vigilance threshold (0 to 1) determines how similar two patterns should be before they are clustered together. The larger the threshold, the less patterns are clustered together and therefore the more clusters will be made. When an input pattern is clustered with an existing or a new cluster centre, its weights are updated. The 'V' weights are set equal to this input vector and the 'W' weights are set to the normalized input vector. While learning this way, the network works as a 'follow the leader' system.

Testing or recall of ART1 is done by feeding a pattern into the input layer and selecting the neuron with maximum response (the winning neuron) using the feedforward weights. The output of the network (i.e. the winning neuron) is the one whose weights are most similar (the smallest Hamming distance) to the input (test) pattern.

Figure 3 shows an ART1 network with 4 inputs and 3 output neurons (clusters); a '4-3' network. Only the weights for one input neuron are shown in the figure.

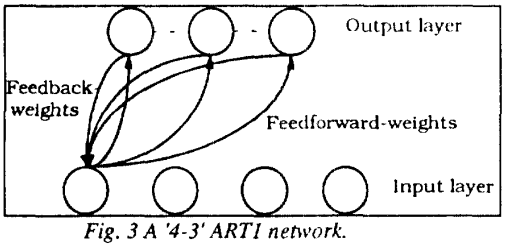

Fig. 3 A '4-3' ART1 network.

The ART1 network can only be used for binary input values. An enhanced version of the algorithm exists, ART2, that can handle continuous valued inputs.

2.5: The Hopfield network [3] [4] [8].

The Hopfield network is a fully connected single layer network. After an input pattern is presented to the network, it will converge by means of a state update rule until it resides in a stable pattern. The state of a neuron is simply its activation (-1 or +1) and the starting state of the network is just the input. The network can only handle bipolar inputs or binary inputs, with a slightly altered update rule. This is the test or recall procedure. The learning or storing is done simply by setting the weights in the network according to $w_{ij}=s_i*s_j$, where w_{ij} is the weight between neurons i and j, and s_i, s_j are the inputs of neurons i and j. Figure 4 shows a 3-neuron Hopfield network.

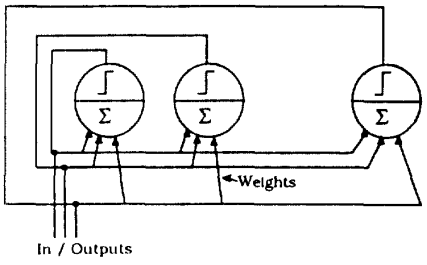

Fig. 4 A 3-neuron Hopfield network.

Usually the weights from a neuron back to itself (i.e. all the weights w_{ii}) are set to zero. When an input is presented to the network, the output of the network will be the stable (trained) pattern that has the minimum Hamming distance from the input pattern. Because of the limited storage capacity, the Hopfield network is not very popular in pattern recognition applications.

In [1] a neural network system is used that consists of a single Hopfield network for every training pattern.

There is virtually no such thing as training time for the Hopfield network. The weights are calculated after the training patterns are presented and are set to their appropriate values. There is no updating of weights. The recall phase does take time because the network has to undergo several convergence steps before the network has found its stable state.

Two 'versions' of the Hopfield network have been implemented. One program is just the ordinary Hopfield network. The other consists of a system of multiple Hopfield network. After the network is trained, it generates as many separate Hopfield networks as there are training patterns and stores one training pattern in every one of them. In the recall phase, the test input is presented to each of the networks. The output (after convergence) of each network is compared with the input pattern and the final output is that output pattern which most closely

(smallest Hamming distance) resembles the input pattern.

2.6: The Bidirectional associative memory (BAM) network.

The Bidirectional Associative Memory is quite similar to the Hopfield network. This type of network has two layers of neurons, which are fully interconnected. In the recall phase, a input (test) pattern is presented to one of the two layers. After the recall (test) phase, the network has come to a stable state so that each of the two layers has a stable pattern as output. The updating of the states of the neurons (-1/+1) is done for the two layers alternatively, which means that there is a bidirectional data flow while the network is converging.

The store (learning) procedure is similar to the Hopfield network except that the BAM network needs inputs to both layers. Figure 9 shows a BAM network which has a layer of three neurons (layer A) and a layer of two neurons (layer B).

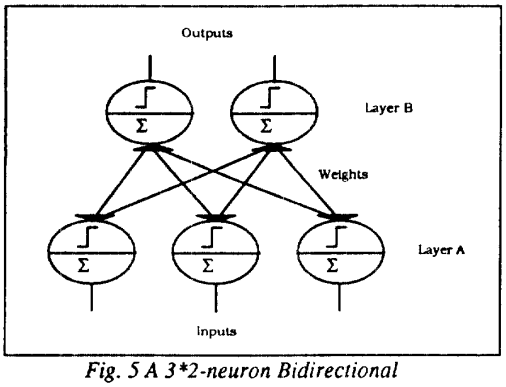

*Fig. 5 A 3*2-neuron Bidirectional Associative Memory.*

For the BAM network, just like the Hopfield network, the storage capacity is probably its main limitation. From [8] it follows that an optimistic value of the maximum storage capacity is: min(m,n), where m,n are the number of neurons in each of the layers. Thus at the most the smallest of m and n patterns can be stored in the network. A more conservative value is [1]: $\sqrt{\min(m,n)}$.

3. Case studies

3.1 Character recognition

A simple comparative study was performed on the neural network simulators described in section 2. The various simulation programs were compared on properties such as speed and generalisation capability (i.e. the performance on distorted input patterns).

The training set consisted of ten digits, '0' to '9' in a 5 x 7 grid as shown below.

```
00100 01110 11111 00010 11111
01100 10001 00010 00110 10000
00100 00001 00100 01010 11110
00100 00010 00010 10010 00001
00100 00100 00001 11111 00001
00100 01000 10001 00010 10001
01110 11111 01110 00010 01110

00110 11111 01110 01110 01110
01000 00001 10001 10001 10001
10000 00010 10001 10001 10001
11110 00100 01110 01111 10001
10001 00100 10001 00001 10001
10001 00100 10001 00010 10001
01110 00100 01110 01100 01110
```

The network architectures which were determined after various trial runs are: a 35x10x10 BP network, a 35x10x10 RBF network, a 2-dimensional 6x6 Kohonen network and a multiple Hopfield network consisting of 10 separate networks. The single Hopfield network and a 35x10 BAM network were not able to learn all the training patterns. Due to the limited storage capacity of these networks only five patterns could be stored.

Fig. 6 shows the cumulative training and test errors as well as the number of incorrectly classified facts for all the test sets. Since only one test set was made for each level of distortion, conclusions from this should be drawn carefully. For the Kohonen, ART1 and Hopfield networks no cumulative errors exist.

pixels distorted:	BP	RBF	Kohonen	ART1& Hopfield
0	0.006	0.00013	-	-
1	0.031 0	0.180 0	- 0	- 0
2	0.023 0	0.396 0	- 0	- 0
3	0.103 0	0.742 1	- 0	- 0
4	0.334 1	1.08 1	- 2	- 0
5	1.04 1	1.60 2	- 4	- 1

Fig. 6 Cumulative errors (top left) and number of incorrectly classified patterns (bottom right) on training and test sets.

It follows that the ART1 and the multiple Hopfield network give the least misclassifications on the test sets. These two networks also showed the best performance in speed; learning took a few seconds as compared to 4 minutes for the RBF network, 10

minutes for the BP network and about 20 minutes for the Kohonen network.
In all, the ART1 showed the best performance on this simple test as it is less computationally expensive than the multiple Hopfield network.

3.2 Further applications

A number of projects are undertaken on the application of neural networks in the Department of Electrical Engineering of the University of Twente. These projects involve handwritten digit recognition, automatic recognition of registration plate numbers of vehicles, identification of the dynamic pattern of signatures, speech recognition, machine condition monitoring, signal source seperation, electric power load forecasting, automatic detection of spike-wave-complexes in EEG-signals, detection of cancer tumours in X-ray photographs, automatic control of a metal sheet bending machine and process parameter identification in plastic production.

4. Conclusions

This paper has described the implementation of neural network simulators in brief. It is obvious from the applictions that neural networks are suitable for a wide range of applications in business, science and engineering.

Acknowledgements

E. Vonk implemented the networks during his 10 weeks work experience training in the Knowledge-based Engineering Systems group, University of South Australia, Adelaide, in 1993. Thanks are due to the department of electrical engineering, University of Twente, for permitting to undertake this research work.

References

[1] Chen, Liang-Chia, Fan, Ju-Yi, and Chen, Yung-Sheng, "A Modified High Speed Hopfield Neural Network And Its Character Recognition System", Proc. ISIC-91, pp. 135-140, (1991).
[2] Mirchandani, Gagan, "On Hidden Nodes for Neural Nets", IEEE Transactions on Circuits & Systems, Vol. 36, No. 5, pp. 661-6, May (1989).
[3] Hopfield, John. J., "Artificial Neural Networks", IEEE Circuits and Devices Magazine, pp. 3-10, (1988).
[4] Hush, D.R. and Horne, B.G., "Progress in Supervised Neural Networks, What'sNew Since Lipmann", IEEE Signal Processing Magazine, pp. 8-39, (1993).
[5] Kohonen, Teuvo, "The Self-Organizing Map", Proceedings of the IEEE, Vol. 78, No. 9, pp.1464-1480, (1990).
[6] Moody, John and Darken, Christian J., "Fast Learning in Networks of Locally-Tuned Processing Units", Neural Computation, Vol.1, pp. 281-294, (1989).
[7] Musavi, M.T. et al., "On the Training of Radial Base Function Classifiers", Neural Networks, Vol. 5, pp. 595-603, (1992).
[8] Zurada, Jacek M., "(Introduction to) Artificial Neural Systems", West Publishing Company, St. Paul, (1992).