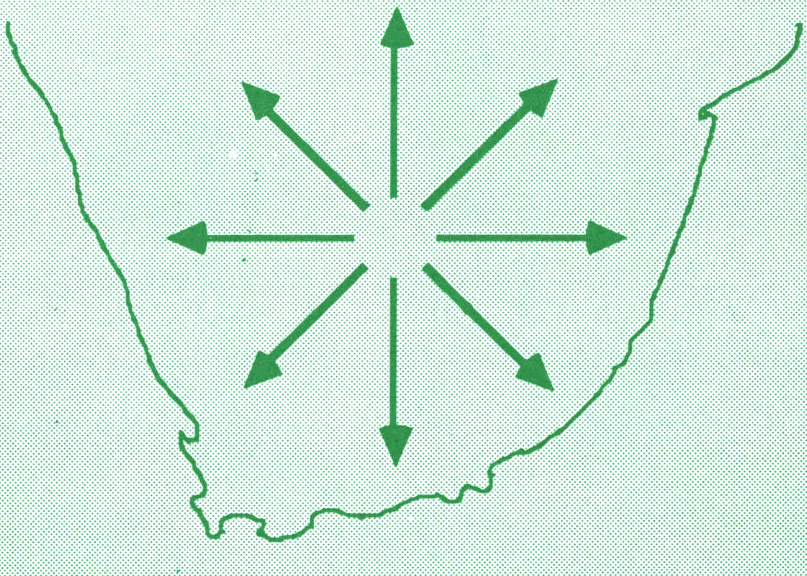
South African Computer Journal
Number 7
July 1992

Suid-Afrikaanse Rekenaar tydskrif
Nommer 7
Julie 1992

SPECIAL ISSUE



7th SA COMPUTER RESEARCH SYMPOSIUM

The South African Computer Journal
An official publication of the South African Computer Society and the South African Institute of Computer Scientists

Die Suid-Afrikaanse Rekenaartydskrif
'n Amptelike publikasie van die Suid-Afrikaanse Rekenaarvereniging en die Suid-Afrikaanse Instituut vir Rekenaarwetenskaplikes

Subscriptions

	Annual	Single copy
Southern Africa:	R45,00	R15,00
Elsewhere	$45,00	$15,00

to be sent to:
Computer Society of South Africa
Box 1714 Halfway House 1685

Computer Science Department, University of Pretoria, Pretoria 0002
Phone: (int)+(012)+420-2504 Fax: (int)+(012)+43-6454 email: dkourie@rkw-risc.cs.up.ac.za

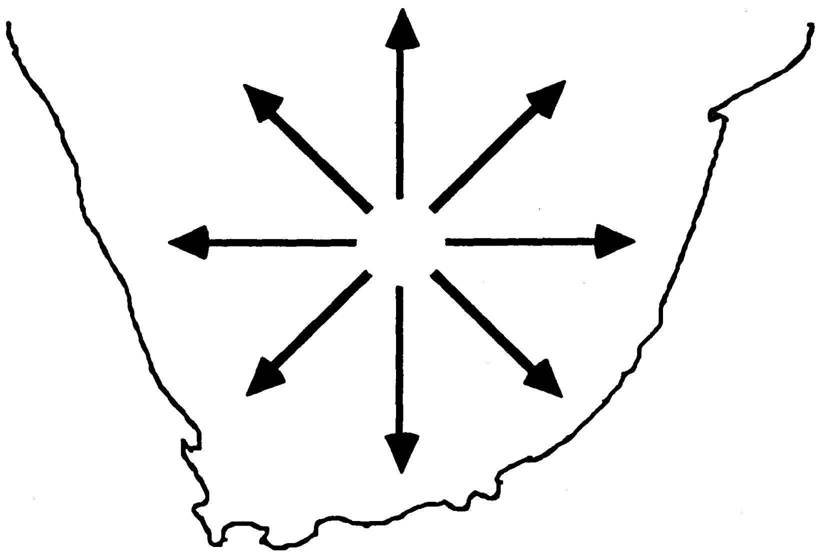
7th Southern African Computer Research Symposium

Karos Indaba Hotel, Johannesburg

1 July 1992

PROCEEDINGS

Guest Editor: Judy M Bishop



PERSETEL

Organised by the SA Institute of Computer Scientists
in association with the Computer Society of SA

Sponsored by Persetel and the FRD

SPECIAL ISSUE - 7th SA COMPUTER SYMPOSIUM

PREFACE

When the first SA Computer Symposium was held at the CSIR in the early eighties, it was unique. There was no other forum at the time for the presentation of research in computer science. In the intervening decade, conferences, symposia and workshops have sprung up in response to demand, and now there are several successful ventures, some into their third or fourth iteration. Each of these addresses a specific topic - for example, hypermedia, expert systems, parallel processing or formal aspects of computing - and attracts a specialised audience, well versed in the subject and eager to learn more. For the main part, the proceedings are informal, and certainly not archival.

SACRS, though, is still unique, in that it deliberately covers a broad spectrum of research in computing, and in addition, seeks to provide a lasting record of the proceedings. To achieve the second aim, we negotiated with the SA Institute of Computer Scientists for the proceedings to form a special issue of the SA Computer Journal, and the copy you have in front of you is the result. The collaboration between the symposium committee and the journal's editorial board placed high standards on the refereeing and final presentation of the papers, to the symposium's benefit, while we were still able to maintain a fresh, audience-oriented approach to the selection of papers.

This is SACJ's first such special issue, and the largest issue (at 145 pages) to date. We hope that it is only the beginning of future such collaborations.

In all 29 papers were received, all were refereed twice, and 19 were chosen for presentation by the programme committee. All the papers were thoroughly revised by the authors on the basis of the referee's comments, and the committee's suggestions aimed at making the material more accessible to a broadly-based audience. Papers had to be new, and not to have been presented elsewhere, a requirement that is still unusual within the SA conference round.

A third goal of SACRS has been to invite keynote speakers, usually from overseas. This year, we are fortunate to present Dr Vinton Cerf, the father of the Internet and a world-renown expert on computer networks. Although his paper is not available for this special issue, it will appear later in SACJ. Through the good offices of Professor Chris Brink of UCT, we also have three other speakers from Germany, Canada and the US adding interest to the event, and two of their papers appear in this issue.

The programme committee originally devised a theme for the symposium - "Computing in the New South Africa". We received several queries as to the meaning of this theme, but unfortunately few papers that addressed it directly. One prospective author went as far as to enquire whether computer research would survive in the new South Africa. Another felt that his work was definitely not in the theme, as it was genuine, old world, basic, theoretical science! Neverthless, there are two papers that consider one of South Africa's key issues, that of language. Others look at the success we have achieved in applying technology to mining, and the future of low-cost operating systems. In all, the mix of papers represents a balance between the theoretical and the practical, the past and the future, all firmly based in the computing of the present.

Organising the symposium has involved the hard work of several people, and I would like to thank in particular

- Derrick Kourie, my co-organiser, and the editor of SACJ for his invaluable advice and hard work throughout the planning and implementation stages;
- Riël Smit, the production editor, for attaining such a high standard in such a short time for so many papers;
- Gerrit Prinsloo and the staff at the CSSA for their efficeint and quite delightfully unfussy organisation;
- Persetel for their very generous sponsorship of R25000, and Tim Schumann for taking a genuine interest in our events;
- the Foundation for Research Development for sponsoring Vint Cerf's visit;
- and finally the Department of Computer Science of the University of Pretoria for providing the ideal working conditions for undertaking ventures of this kind, and especially Roelf van den Heever for his unfailing encouragement and support.

Judy M Bishop
Organising Chairman, SACRS 1992
Guest Editor, SACJ Special Issue

Referees

The journal draws on a wide range of referees. The following were involved in the refereeing of the papers selected for this special issue. Their role in certifying the papers and their contribution to enhancing the quality of papers is sincerely appreciated.

John Barrow	UNISA
Ronnie Becker	University of Cape Town
Danie Behr	University of Pretoria
Sonia Berman	University of Cape Town
Liesbeth Botha	University of Pretoria
Theo Bothma	University of Pretoria
Chris Brink	University of Cape Town
Peter Clayton	Rhodes University
Ian Cloete	Stellenbosch University
Antony Cooper	CSIR
Elise Ehlers	Rand Afrikaanse University
Quintin Gee	DALSIG
Andy Gravell	University of Southampton
Wendy Hall	University of Southampton
David Harel	The Weizmann Institute of Science
Scott Hazelhurst	University of the Witwatersrand
Derrick Kourie	University of Pretoria
Willem Labuschagne	UNISA
Doug Laing	ISM
Dave Levy	University of Natal
Graham McLeod	University of Cape Town
Hans Messerschmidt	University of the Orange Free State
Deon Oosthuizen	University of Pretoria
Ben Oosthuysen	University of Pretoria
Neil Pendock	University of the Witwatersrand
D Petkov	University of Natal
Martin Rennhackkamp	Stellenbosch University
Cees Roon	University of Pretoria
Jan Roos	University of Pretoria
John Shochot	University of the Witwatersrand
Morris Sloman	Imperial College, London
Riel Smit	University of Cape Town
Pat Terry	Rhodes University
Walter van den Heever	University of Pretoria
Lynn van der Vegte	University of Pretoria
Herman Venter	University of Fort Hare
Herna Viktor	Stellenbosch University

Animating Neural Network Training

Etienne van der Poel Ian Cloete

Department of Computer Science, University of Stellenbosch, Stellenbosch 7600, South Africa

Internet: ian@cs.sun.ac.za

Abstract

In Artificial Neural Network (ANN) simulation it is usually necessary to examine the behaviour of the ANN and detect problems if they occur. However, due to the large volume and high dimensionality of data generated during ANN simulation, interpretation of results is difficult. By using visualisation techniques, such as simple animated faces, problems that occur during the training of a backpropagation neural network can be detected and analyzed. The dynamic behaviour of the network is also better understood when using animation as part of the visualization process.

Keywords: *neural networks, backpropagation, visualization*
Computing Review Categories: *I.3.4, I.5.2*

Received March 1992, Accepted April 1992

1 Introduction

Artificial neural networks (ANN) are based on the most powerful problem solving device known - the human brain - and provides an alternative form of computing from the well-known von Neumann based computing.

An ANN consists of a network of interconnected processing elements, or neurons. Each neuron has one or more weighted inputs from other neurons or from sources external to the ANN. These weighted inputs are added together, and then passed through a non-linear function to produce the output of the neuron. A fully connected feedforward neural network consists of one or more layers of neurons, where the output of a neuron serves as input to all the neurons in the layer above it. To use the ANN, an input is presented to the first layer. The outputs of each layer of neurons are then calculated before those of the layer above it, starting with the lowest layer. Finally, the top layer produces the ANN's output [6].

An ANN is trained by adjusting the weights on the neuron inputs. It can be trained to produce a vector mapping where the exact mapping is not known, but embedded in a set of examples that represents the function. In fact, it has been proven that a feedforward neural network can exactly implement any continuous function $f: [a, b]^n \rightarrow R^m$ [3].

Artificial neural networks have been used in applications ranging from optimization and classification through to signal processing and speech analysis.

Due to the large volume and high dimensionality of data generated during an artificial neural network simulation, we are investigating visualization as an alternative mechanism to understand and analyze the behaviour of an ANN. This paper presents the results of visualization experiments conducted using a batch-update Backpropagation (BP) neural network. We show that by representing the neuron data with simple faces [1] understanding of neural network behaviour is greatly enhanced because it enables the simultaneous display of all BP variables of a particular neuron, while animation of the faces clearly show significant stages of the optimization process. Furthermore, the faces can also serve as an educational tool to teach the principles of ANN training and operation.

The organisation of the paper is as follows: Firstly a brief overview of the BP algorithm and the use of the faces is given. Then the test domain and mapping of BP variables onto facial features are presented, followed by an exposition of significant patterns observed.

2 The Backpropagation Algorithm

The BP algorithm trains a multilayer feedforward neural network, using supervised training. The training set consists of P pairs of input and output vectors, $(x_1, x_2, \ldots x_N)$ and $(t_1, t_2, \ldots t_M)$, where x_i is the input to input neuron i, and t_j is the desired output of output neuron j. There are N input neurons and M output neurons.

Each neuron i, in the network has J_i input connections i_{ji}, each of which has an associated weight w_{ji}. Each neuron also has an internal bias θ_i, a net input value net_i, an output or activation value o_i, and an activation function $f_a(net_i)$. The following relationships are defined between these variables:

$$net_i = \sum_{j}^{J_i} w_{ji} i_{ji} + \theta_i \qquad (1)$$

$$o_i = f_a(net_i) \qquad (2)$$

$$f_a(net_i) = \frac{1}{1 + e^{-net_i}} \qquad (3)$$

The algorithm has a feedforward phase, during which the network outputs are calculated for the current training pattern, followed by an error backpropagation phase, during which the error in the network's output is propagated backwards, layer by layer, to adapt the weights in the network connections, and in so doing minimizing the total summed squared error E, of the network output. If t_{pm} and o_{pm} are, respectively, the desired output and actual output of output neuron m when training pattern p is presented to the network, then E is defined as:

$$E = \frac{1}{2}\sum_{p}^{P}\sum_{m}^{M}(t_{pm} - o_{pm})^2 \qquad (4)$$

Each neuron also has several variables which are used to keep track of the neuron's state during the error back-propagation phase of the algorithm. Each neuron has an error value e_i, which is a measure of a particular neuron's contribution to the total error. The neuron delta value δ_i, a measure of the changes in the neuron weights, is calculated from e_i and the net_i value, which in turn is used to calculate the weight error derivatives, wed_{ji}. A gain constant η, is used to determine the rate at which the neuron weights change (i.e. the learning rate), while a momentum constant α, is used to smooth the weight changes. The Δw_{ji}, which is the value with which weight w_{ji} should change, is calculated by summing wed_{ji} over all the training patterns, multiplying this with η, and adding α times the previous Δw_{ji}. The Δw_{ji} is then added to w_{ji} to form the new w_{ji}. The bias error derivative bed_i and $\Delta\theta_i$ are used in a similar manner to update the θ_i value [6].

Once all the training patterns have been presented to the network a single *epoch* is said to have been completed. The algorithm has to cycle through several epochs to complete the training of the network [4, 6].

3 Visualizing Neural Network Data

In recent literature visualization of data has received much attention [2, 5]. The problem is to present data in such a way that the human creative and analytic processes can be utilized effectively. This requires a method of communication with a high bandwidth and an effective interface. Scientific visualization uses the human vision system and computer-generated images. With colour, form, intensity, transparency, and many other techniques, properly prepared images can present a massive amount of information in a short time. Not only does a static display present a lot of information, but the dynamic behaviour of an ANN can also be inspected using animation techniques. Human perception mechanisms are quite sensitive to changes in images, usually resulting in a firmer understanding of problems.

The faces-method is a visualization technique not yet utilized for neural networks. It was designed to detect clustering in multidimensional vectors by mapping the same variables in each vector onto the same facial feature. If one or more faces then appear similar, their corresponding vectors would also be "similar".

In order to visualize BP training, we chose to represent each neuron in the network with a single face, and to map the neuron data values onto the facial features by scaling each variable to a range suitable for the feature. The facial expression then gives a representation of the state of the neuron and its internal variables. Furthermore, by animating faces the changes in facial expressions make the observation of the dynamic nature of the network possible.

In order to successfully analyze the behaviour of the network, we need to be able to:

- Recognize a significant pattern in a static display at a particular point during the simulation process.
- Observe significant patterns in the dynamic behaviour of the neural network using animation of the faces.

Due to the way in which the BP algorithm operates, it makes sense to perform a static analysis at the end of a simulation run, at the end of a particular epoch, or after processing a particular example. Similarly, animation of changing values at the end of each epoch over a range of epochs is suitable for observing significant changes in the optimization process, because at that point variables are updated and some are subsequently reset for the next epoch. In the same way animation of changes over all examples shows observable trends during an epoch before variables are reset for the next epoch.

4 The Features of the Face

The facial features each have a certain range within which they operate. These ranges are either $[0, +]$, zero to a maximum positive, or $[-, +]$, a maximum negative to a maximum positive. This means that certain features can only display positive values, while others can display both positive and negative values. For the weight changes Δw_{ji} and the weight error derivatives wed_{ji} (mapped onto the eyes and ears, respectively), a single number was computed to obtain an average absolute measure, indicative of the magnitude of change in these variables. Table 1 shows the features, the aspect of a feature that is adjustable, its range, as well as the data values mapped onto the feature. The weights are mapped onto the hair, with the first weight

Table 1. The facial feature mapping

Feature	Change	Range	Data mapped
mouth	smile	$[-, +]$	net input net_i
mouth	width	$[0, +]$	output o_i
hair	length	$[-, +]$	weights w_{ji}
eyebrows	slant	$[-, +]$	error e_i
nose	width	$[-, +]$	delta δ_i
eyes	radius	$[-, +]$	$(\sum_i^k \mid \Delta w_{ji} \mid)/k$
ears	size	$[0, +]$	$(\sum_i^k \mid wed_{ji} \mid)/k$

being the first hair, and so on. The eyes may also change colour, from white to black, or vice versa, when any of the Δw_{ji} changes its sign from the previous iteration. This change of colour is optional and may be switched off.

The mapping of variables onto features is done by linearly scaling the variables to the operable range of the feature. The scaling transformation assures that a zero data value maps onto a zero feature value, that the same scale is used for both the positive and negative values of a feature, and that the same scale is also used for the same feature on all the faces, thereby maintaining relative proportions between the negative and positive sides of a feature as well as between faces. Figure 1 displays three faces, showing the minimum, zero, and maximum feature values.

In subsequent figures, the first hidden layer neurons are represented by the top row of faces, and the output neurons are represented by the bottom row of faces, as in Figure 2.

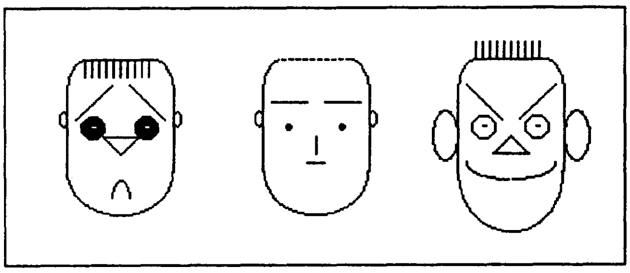


Figure 1. Minimum, zero, and maximum faces

5 The Decision Borders

The purpose of a first hidden layer neuron is to form a decision border [4] in the example space, which is a hyperplane separating the two classes. The decision border is defined as those points in the example space where $f_a(net_{ji}) \approx 0.5$, i.e. midway between the minimum and maximum output for the neuron. This occurs when $net_i \approx 0$. In two-dimensional space a decision border occurs where:

$$w_{1i}i_{1i} + w_{2i}i_{2i} + \theta_i \quad \approx \quad 0 \tag{5}$$

A set of decision borders that are close to ideal for our training set are those shown in Figure 3, where the three lines form regions in such a way that there are examples of no more than one class inside each of these regions.

The decision border display shows only the hidden layer decision borders. The decision border equations are arranged top to bottom, with the topmost equation being that of the first hidden neuron, which corresponds with the top left face, the second equation is the second hidden neuron, and corresponds with the top middle face, and so on.

6 The Test Domain

A small, fully connected feedforward network consisting of a hidden layer of three neurons and an output layer of two neurons, was trained using an artificial training set. The training set consists of 85 two-dimensional vectors divided into two classes. There are 55 vectors of one class, and 30 of the other. For class 0, the first output neuron should have a value of 1, while the other neuron has a value of 0. For class 1, these values should be reversed. The vectors have been chosen in such a way that at least three hidden layer neurons are required to separate them into two distinct regions, i.e. class 1 inside an enclosed region, and class 0 outside the enclosed region. Some vectors have been chosen to be very close to the ideal decision borders, making the problem more difficult.

Two-dimensional input vectors were chosen so as to be able to display the training set graphically and to relate the faces to corresponding decision borders.

For the experiments different initial weights and biases, as well as different momentum and gain terms were used with the same training set. Five training runs are presented, one successful, and four unsuccessful; they all use the same training set.

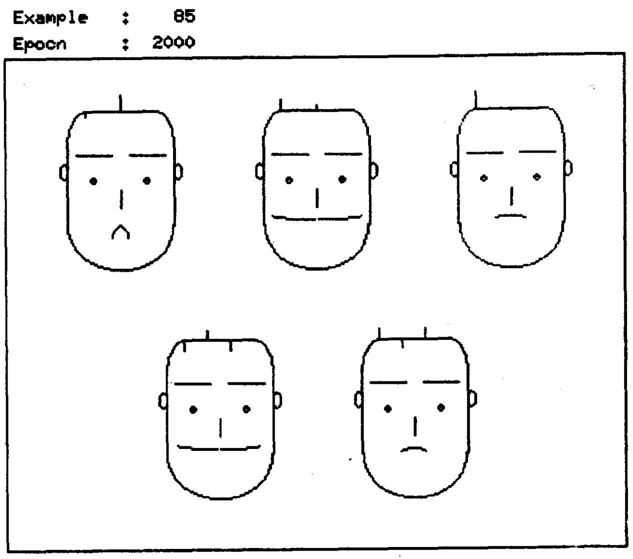

```
Example  :   85
Epocn    :  2000
```

Figure 2. Faces for training run 1.

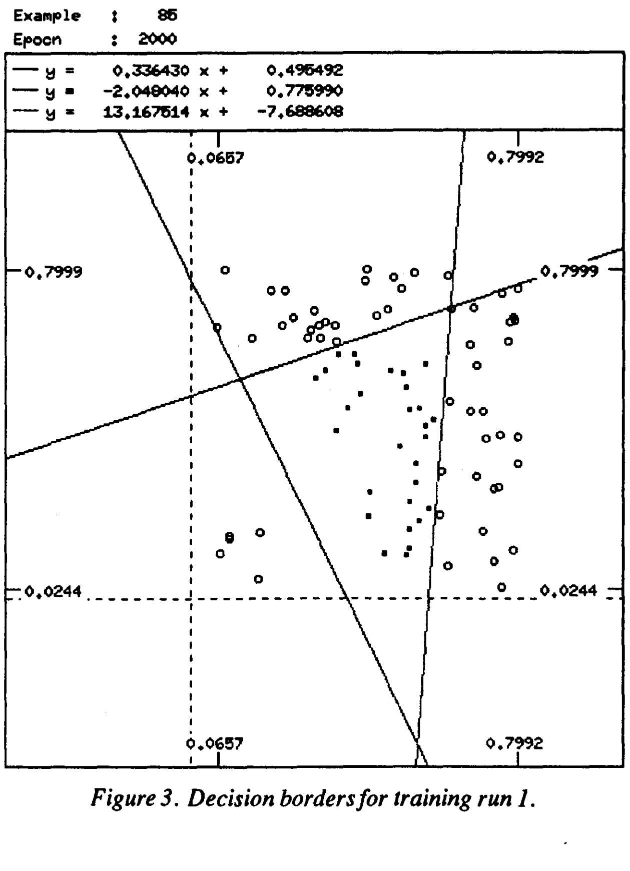

```
Example  :   85
Epocn    :  2000
— y =    0,336430 x +    0,496492
— y =   -2,048040 x +    0,775990
— y =   13,167614 x +   -7,688608
```

Figure 3. Decision borders for training run 1.

```
Example    :    85
Epocn      : 25068
```

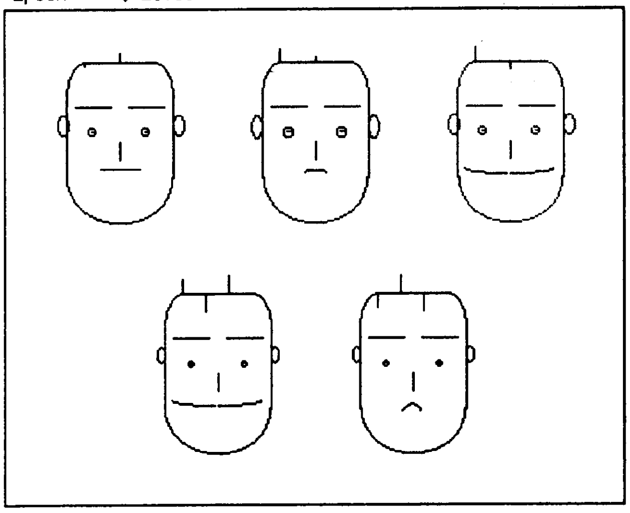


Figure 4. Faces for training run 2.

```
Example    :    85
Epocn      : 25069
```

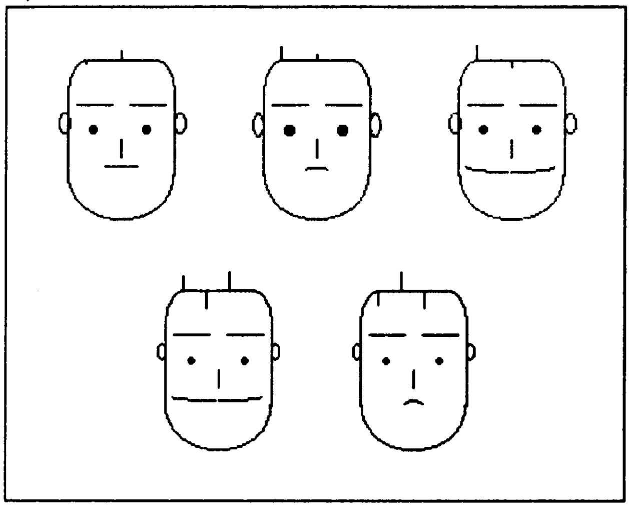


Figure 5. Faces for training run 2.

```
Example    :    86
Epocn      : 25069
```

```
— y =   0,331482 x +   0,112061
— y =  -2,290725 x +   1,720960
— y =   1,981273 x +  -0,196409
```

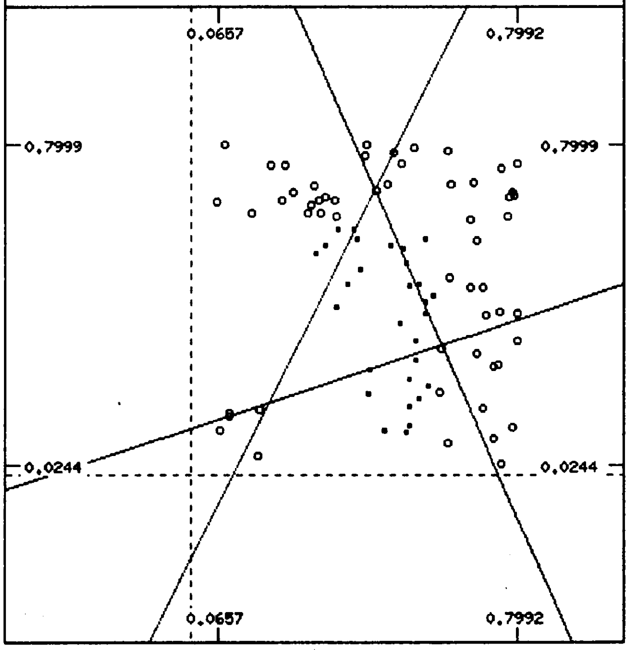


Figure 6. Decision borders for training run 2.

7 Significant Patterns Observed

By using faces we wanted to answer several questions about network performance and evaluate how well the technique shows recognizable patterns[1]. These questions are discussed and a brief explanation given.

The most important question to answer is probably whether the network was *successfully trained.* Successful training occurs when E is close to zero. From the experiments this question is easily resolved by observing the slant (or changes in slant) of the eyebrows of the output neurons when animating all examples in an epoch. Because the error value for a particular neuron is mapped to the eyebrows in a face, horizontal eyebrows for all output neurons and all examples indicate successful training. Due to the human eye's sensitivity to movement, eyebrows indicative of ongoing error activity are easily noticed. Figure 2 shows the faces for successful training, while Figure 3 shows decision borders created by the hidden neurons. In the same way it is also easily observed when the network is approaching a stable solution – the ears, eyebrows, eyes, and nose approach their zero positions and little change in those features occur.

Conversely, unsuccessful training due to *instability* in the optimization process is also recognizable. The question is whether the criterion function is converging to an acceptably low error value for all training examples. Two parameters of a BP network, the momentum and gain, play an important role in this question. Incorrectly chosen values could cause instability of the optimization process, and is therefore important to detect. The continually changing sign of the Δw_{ji} after every epoch indicates an oscillation in the iteration. Figures 4 and 5 show snapshots of this phenomenon. Note the eyes, which change between black and white. The decision borders of the hiddens, as shown in Figure 6, move to an fro with every epoch without improving the classification accuracy.

Instability is not the only cause for unsuccessful training. Figures 7 to 16 show the faces and decision borders of hidden neurons found for two other unsuccessful training runs. Animation over the last few epochs of these runs (Figures 7 and 15) shows a pattern occurring in both, namely that the optimization process cannot escape from a *local minimum.* This phenomenon is characterized by non-zero error values (slanted eyebrows) at the output neurons, but no changes in the size of the eyes and ears. The reason is that weight changes have ceased ("zero size eyes") while the derivative of the criterion function with respect to the weights approach zero ("zero size ears").

Other interesting patterns are also observed. In the first

[1]It should be noted that the human eye is very sensitive to movement, and for that reason animated displays are effective for showing dynamic changes. This also means that the single frames shown in this article unfortunately do not convey the complete effect of animation.

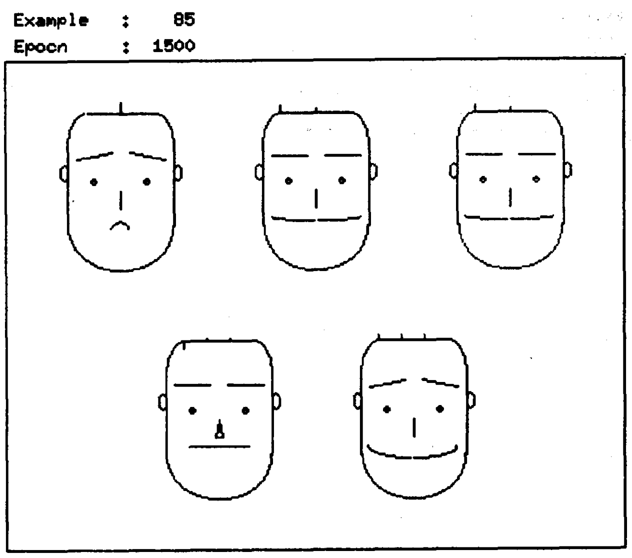


Figure 7. Faces for training run 3.

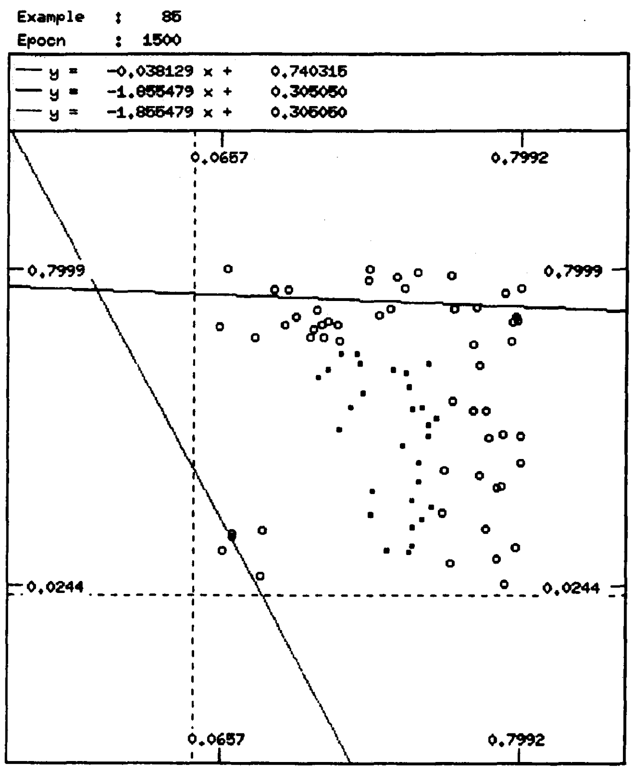


Figure 8. Decision borders for training run 3.

run two of the hidden neurons show *identical faces* for all examples, as seen in Figure 7. The decision borders of these two neurons discriminate for identical sets of examples (duplicate function [7]) and resulted in an inadequate classifier. Two decision borders of the hidden neurons are identical, as in Figure 8.

In the second simulation run (Figures 9 to 16) animation of the three hiddens shows much activity (e.g. Δw_{ji}'s as shown in the eyes) while the decision border traverses the example area. The activity decreases markedly after epoch 3 when the decision borders have moved away from the training set. After that point the absolute values of the weights start to increase markedly, forcing the hidden neurons' activation values closer to zero. The derivative of the activation function approaches zero because the activation values become smaller and smaller, forcing the delta of each hidden neuron to zero, thus eventually no further weight changes for the hiddens are possible. This means that the examples are classified incorrectly. Note that, at epoch 100 (Figure 15), the mouths of the hidden neurons all show that the activation level is almost zero and the net input is large for all examples. This pattern thus shows that none of the hiddens participate in the discrimination task for any example (hiddens are both inactive and stray [7]) and have effectively turned off.

Figures 17 to 20 for the third simulation run show a result where one hidden neuron does not discriminate between any two examples – the left face has a small downward smile for all examples as above (inactive hidden neuron [7]). The dynamic network behaviour further shows clearly during the run (as for the second run above) whether each hidden neuron participates in the discrimination task. Figures 17 to 20 show the onset of discrimination for the middle face (larger eyes). At epoch 2726 (Figure 19) the corresponding decision border has moved into the example area. The same pattern was observed for the rightmost face at epoch 2763 when its corresponding decision border reaches the example region. In both cases the left face remains non-discriminating.

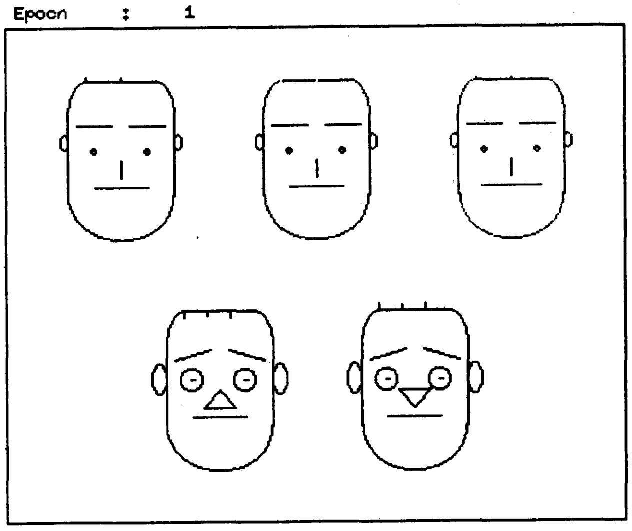
8 Conclusion

The faces-method as a visualization technique displays the characteristics of individual BP neurons concisely, because it allows the display of all BP variables per neuron simultaneously. The faces-method also enables the identification of significant dynamic behaviour patterns during BP training.

The faces are capable of showing characteristic states of the network and individual neurons by means of certain identifiable facial expressions, and in the positions and sizes of the individual features. Non-discriminating neurons show clearly in an animated display, because features of neurons participating in the discrimination task change, but the faces of the inactive neurons do not. Identical faces clearly show that neurons are duplicating their function. Oscillations without improving network performance have also been detected in the changing colour of the eyes, while the eye size remains the same.

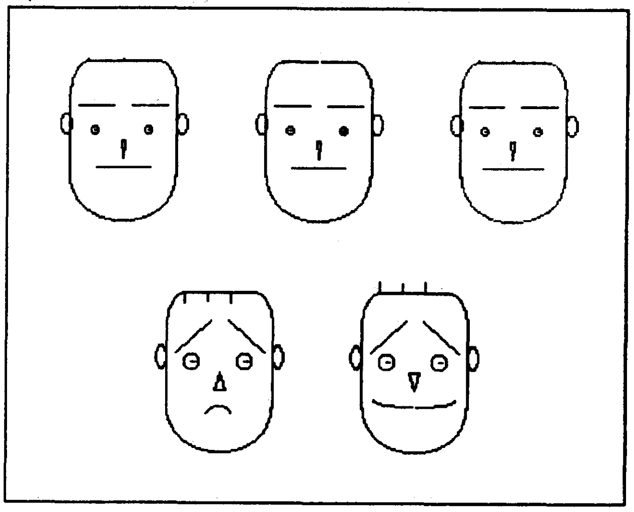
The effectiveness of the faces-method as a visualization technique is, admittedly, a subjective matter. Because the number of neurons for which the technique can be used simultaneously is currently limited by the size of the display and human ability to keep track of varying complex patterns, we are investigating other visualization techniques, as well as alternative ways to select interesting patterns for more detailed observation.

In conclusion, the faces-method provides a novel, yet natural way of visualizing multi-dimensional neural network data which allows significant static and dynamic behaviour patterns to be recognized. As an educational tool, the faces would be a useful way to teach students the principles of neural network operation and training.



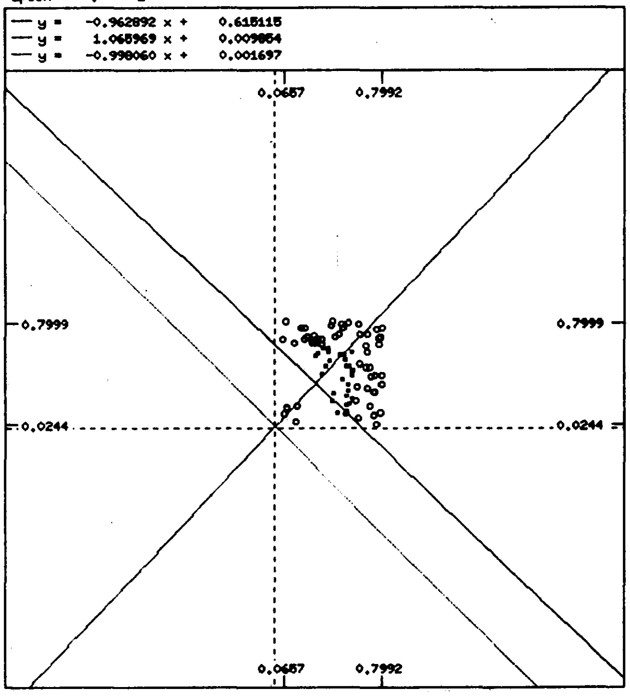
```
Example  :   85
Epocn    :    1
```

Figure 9. Faces for training run 4.



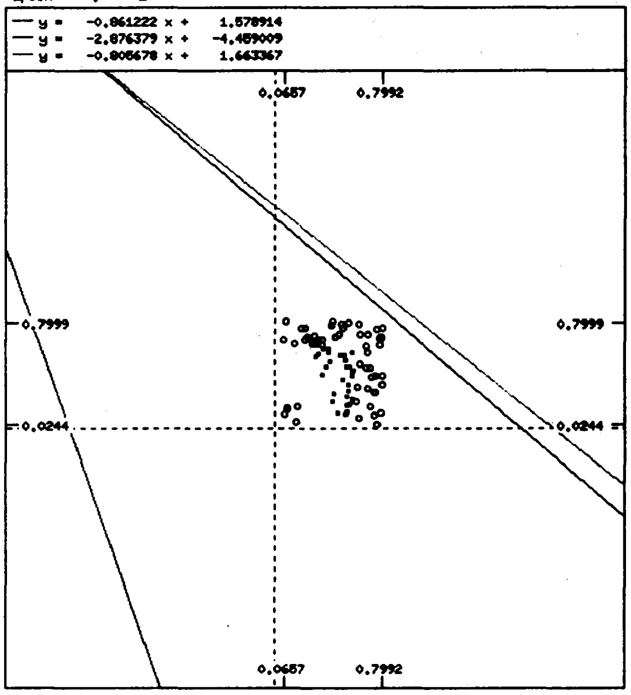
```
Example  :   85
Epocn    :    2
```

Figure 11. Faces for training run 4.



```
Example  :   85
Epoon    :    1
— y =  -0.962892 x +   0.615115
— y =   1.065969 x +   0.009854
— y =  -0.998060 x +   0.001697
```

Figure 10. Decision borders for training run 4.



```
Example  :   85
Epoon    :    2
— y =  -0.861222 x +   1.578914
— y =  -2.876379 x +  -4.459009
— y =  -0.805678 x +   1.663367
```

Figure 12. Decision borders for training run 4.

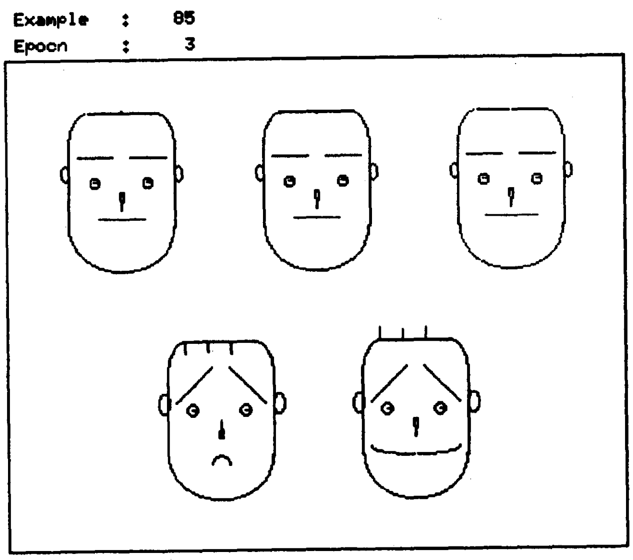


Figure 13. Faces for training run 4.

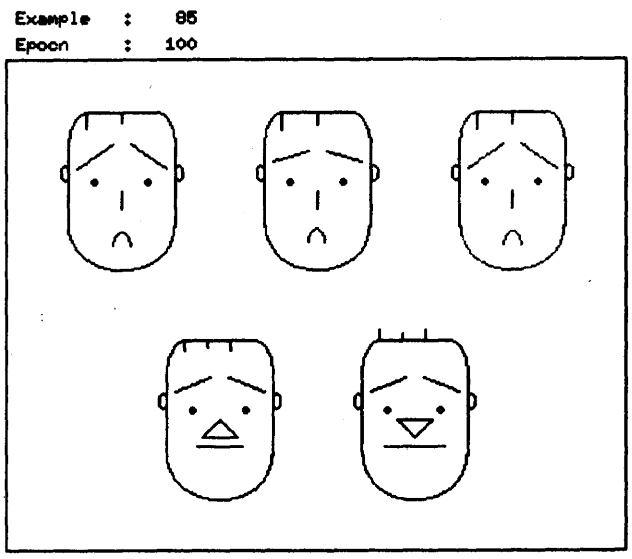


Figure 15. Faces for training run 4.

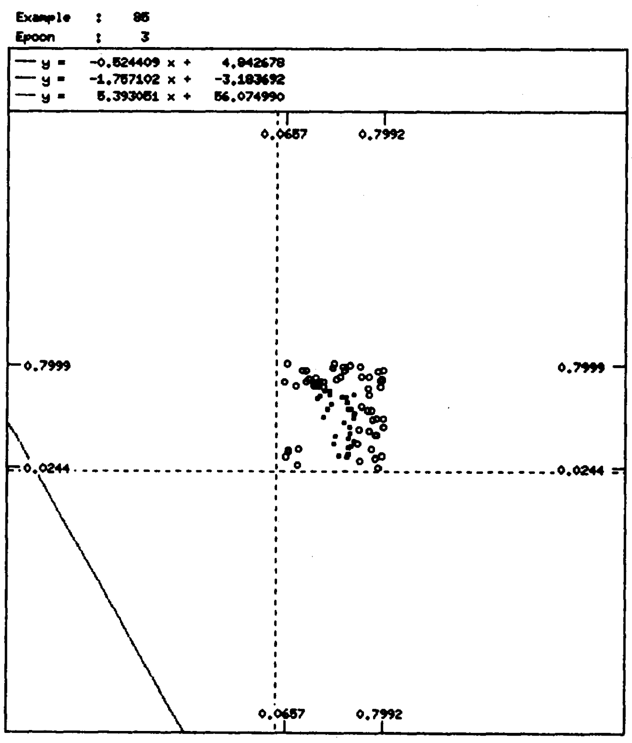


Figure 14. Decision borders for training run 4.

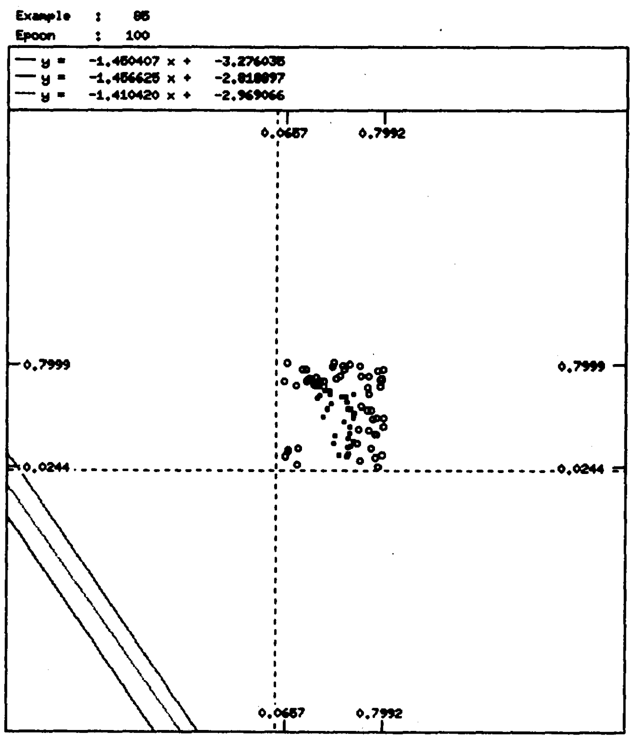


Figure 16. Decision borders for training run 4.

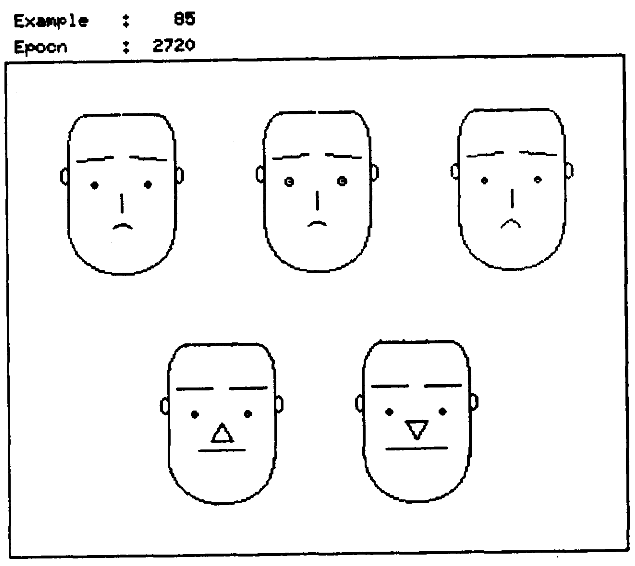


Figure 17. Faces for training run 5.

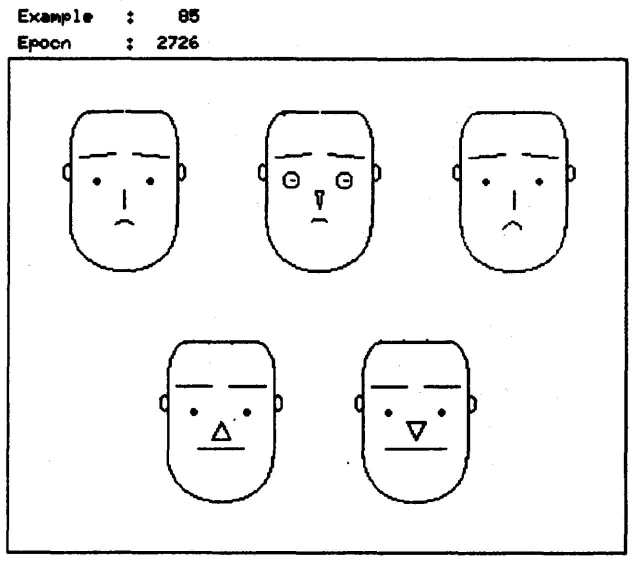


Figure 19. Faces for training run 5.

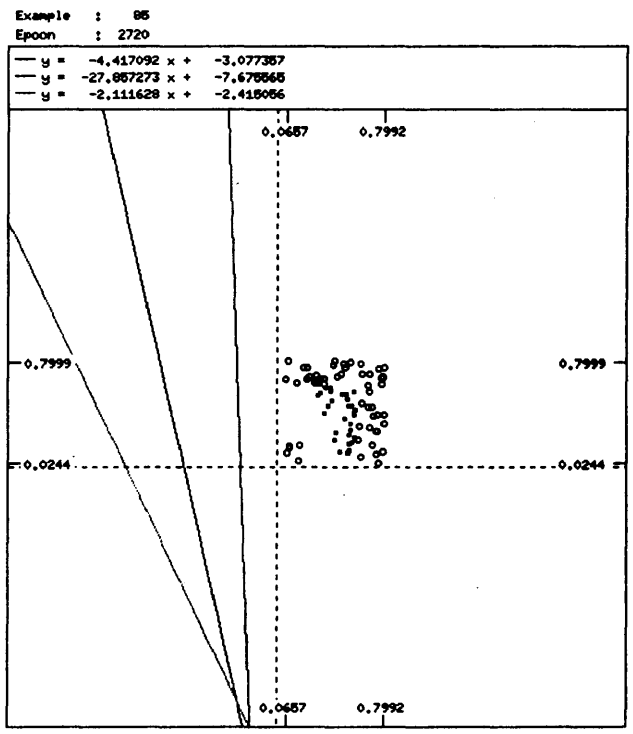


Figure 18. Decision borders for training run 5.

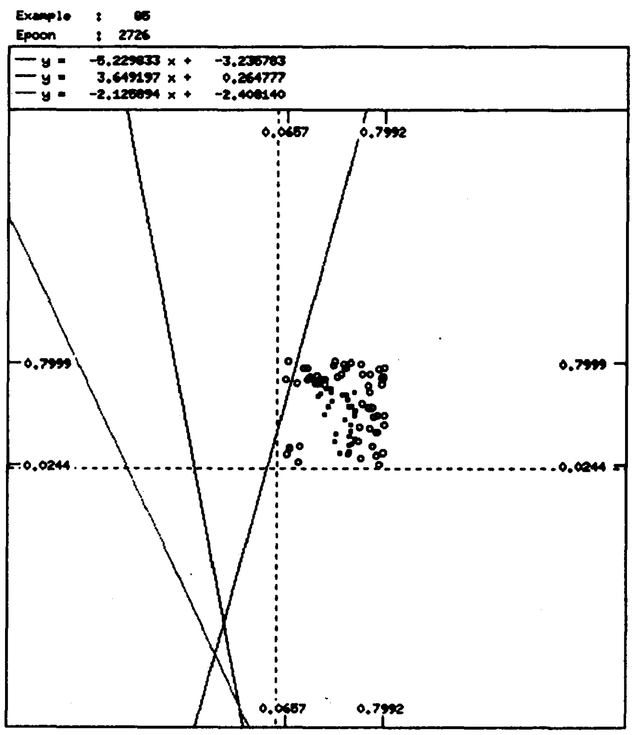


Figure 20. Decision borders for training run 5.

References

1. H Chernoff. 'The Use of Faces to Represent Points in k-Dimensional Space Graphically'. *Journal of the American Statistical Association*, **68**(243):362–368, June 1973.
2. K A Frenkel. 'The Art and Science of Visualizing Data'. *Communications of the ACM*, **31**(2):111–121, February 1988.
3. R Hecht-Nielsen. *Neurocomputing*. Pages 122–124. Addison-Wesley, 1990.
4. R P Lipman. 'An Introduction to Computing with Neural Nets'. *IEEE ASSP Magazine*, **4**(2):4–22, April 1987.
5. G M Nielson. 'Visualization in Scientific and Engineering Computation'. *IEEE Computer*, **24**(9):58-66, September 1991.
6. D E Rumelhart, and J L McClelland. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Volume 1: Foundations. Pages 118–130. MIT Press, 1986.
7. L F A Wessels, E Barnard, and E van Rooyen. 'The Physical Correlates of Local Minima'. *South African Computer Journal*, **5**:22–27, September 1991.

Notes for Contributors

The prime purpose of the journal is to publish original research papers in the fields of Computer Science and Information Systems, as well as shorter technical research papers. However, non-refereed review and exploratory articles of interest to the journal's readers will be considered for publication under sections marked as Communications or Viewpoints. While English is the preferred language of the journal, papers in Afrikaans will also be accepted. Typed manuscripts for review should be submitted in triplicate to the editor.

Form of Manuscript

Manuscripts for *review* should be prepared according to the following guidelines.

- Use wide margins and $1\frac{1}{2}$ or double spacing.
- The first page should include:
  - title (as brief as possible);
  - author's initials and surname;
  - author's affiliation and address;
  - an abstract of less than 200 words;
  - an appropriate keyword list;
  - a list of relevant Computing Review Categories.
- Tables and figures should be numbered and titled. Figures should be submitted as original line drawings/printouts, and not photocopies.
- References should be listed at the end of the text in alphabetic order of the (first) author's surname, and should be cited in the text in square brackets [1, 2, 3]. References should take the form shown at the end of these notes.

Manuscripts accepted for publication should comply with the above guidelines (except for the spacing requirements), and may be provided in one of the following formats (listed in order of preference):

1. As (a) LaTeX file(s), either on a diskette, or via e-mail/ftp – a LaTeX style file is available from the production editor;
2. In camera-ready format – a detailed page specification is available from the production editor;
3. As an ASCII file accompanied by a hard-copy showing formatting intentions:
   - Tables and figures should be on separate sheets of paper, clearly numbered on the back and ready for cutting and pasting. Figure titles should appear in the text where the figures are to be placed.
   - Mathematical and other symbols may be either handwritten or typed. Greek letters and unusual symbols should be identified in the margin, if they are not clear in the text.

   Further instructions on how to reduce page charges can be obtained from the production editor.
4. In a typed form, suitable for scanning.

Charges

Charges per final page will be levied on papers accepted for publication. They will be scaled to reflect scanning, typesetting, reproduction and other costs. Currently, the minimum rate is R20-00 per final page for LaTeX or camera-ready contributions and the maximum is R100-00 per page for contributions in typed format.

These charges may be waived upon request of the author and at the discretion of the editor.

Proofs

Proofs of accepted papers in categories 3 and 4 above will be sent to the author to ensure that typesetting is correct, and not for addition of new material or major amendments to the text. Corrected proofs should be returned to the production editor within three days.

Note that, in the case of camera-ready submissions, it is the author's responsibility to ensure that such submissions are error-free. However, the editor may recommend minor typesetting changes to be made before publication.

Letters and Communications

Letters to the editor are welcomed. They should be signed, and should be limited to less than about 500 words.

Announcements and communications of interest to the readership will be considered for publication in a separate section of the journal. Communications may also reflect minor research contributions. However, such communications will not be refereed and will not be deemed as fully-fledged publications for state subsidy purposes.

Book reviews

Contributions in this regard will be welcomed. Views and opinions expressed in such reviews should, however, be regarded as those of the reviewer alone.

Advertisement

Placement of advertisements at R1000-00 per full page per issue and R500-00 per half page per issue will be considered. These charges exclude specialized production costs which will be borne by the advertiser. Enquiries should be directed to the editor.

References

1. E Ashcroft and Z Manna. 'The translation of 'goto' programs to 'while' programs'. In *Proceedings of IFIP Congress 71*, pp. 250–255, Amsterdam, (1972). North-Holland.
2. C Bohm and G Jacopini. 'Flow diagrams, turing machines and languages with only two formation rules'. *Communications of the ACM*, 9:366–371, (1966).
3. S Ginsburg. *Mathematical theory of context free languages.* McGraw Hill, New York, 1966.

South African Computer Journal

Number 7, July 1992
ISSN 1015-7999

Suid-Afrikaanse Rekenaartydskrif

Nommer 7, Julie 1992
ISSN 1015-7999

Contents