ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

Neural Networks which identify Composite Factors

Donald MacDonald, Darryl Charles and Colin Fyfe
Department of Computing and Information Systems
Applied Computational Intelligence Research Unit
The University of Paisley
Scotland
e-mail: mcdo-ci0,char-ci0,fyfe-ci0@paisley.ac.uk

ABSTRACT

We investigate the use of an artificial neural network to form a sparse distributed representation of the underlying factors in data sets. We extend the previously proposed [1] network so that it may identify composite causes in data sets by creating a hierarchical network. We use the network as a means of identifying individual faces when the network is trained on a mixture of faces and show both analytically and through experiments how noise allows us to find precisely the factors without prior assumptions of the number of factors.

1. Introduction

We have previously [1] extended a negative feedback network which had been shown [2] to be capable of performing a Principal Components Analysis (PCA) of the input data so that it now finds an optimal "sparse coding" of the input data. A sparse coding (Figure 1) is one in which the dimensionality of the code is not necessarily less than the input dimensionality but each instance of the code uses a much smaller number of non-zero values at any one time. This is biologically interesting since Barlow [3] hypothesizes that early sensory processing serves to transform the highly redundant sensory signal into a more efficient factorial code. We note in this paper that such codings are most appropriate when the data set is itself composed of factors which have a sparse distribution.

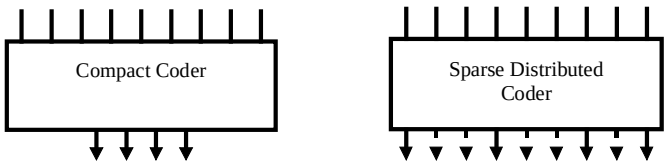


Fig. 1 The compact coder transforms the data by reducing the dimensionality of the inputs, whereas the other maintains the dimensionality of the data but sparsifies their representation

The benchmark problem, proposed by Foldiak[4], is that of identifying the individual bars from a set of input data which contains a mixture of bars (Figure 2 shows examples). The data set consists of an 8*8 square grid containing a random mixture

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

of horizontal and vertical bars. The input value $x_i = 1$ if a square is black and is 0 otherwise and the black bars are chosen randomly such that each of the 16 possible bars are chosen with a fixed probability of 1/8 independent of each other. Note that there are (nearly) 2^{16} possible patterns in the data set, compared with the 2^{64} patterns which could be represented on the 8*8 grid.

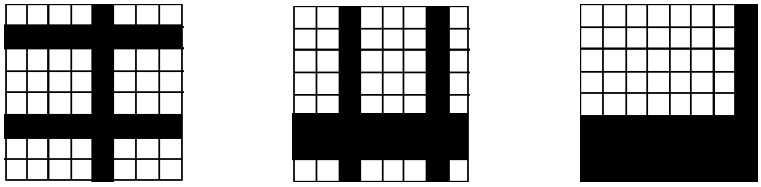


Figure 2 Samples of input data presented to the network

Now we wish to represent this data set, with no loss of information, but to do so in a way which reveals the underlying causes of the data set. At one extreme, we might have a totally distributed representation in which each of the 2^{16} possible patterns is identified when a particular output neuron fires i.e. we have almost 2^{16} output neurons with only one firing at any one time. The opposite extreme is the compact coder such as the PCA coder which gives us a global compression of the data. We aim for a middle way: the bars are the underlying causes of the input data and so we wish the output neurons to identify precisely one of the horizontal or vertical bars. Thus the ensemble of output neurons will, acting together, be able to recreate the input pattern. We will require 16 output neurons to represent the 16 underlying causes of the data set.

2. The Negative Feedback Network

The basic PCA network[2] is described by equations (1)-(3). Let us have an N-dimensional input vector at time t, **x(t),** and an M-dimensional output vector, **y,** with w_{ij} being the weight linking input *j* to output *i* and η the learning rate. Then the activation passing and learning is described by

$$y_i = \sum_{j=1}^{N} w_{ij} x_j , \qquad \forall i \tag{1}$$

$$x_j(t+1) = x_j(t) - \sum_{i=1}^{M} w_{ij} y_i \tag{2}$$

$$\Delta w_{ij} = \eta x_j(t+1) y_i \tag{3}$$

We note that this algorithm is equivalent to Oja's Subspace Algorithm [5] since

$$\Delta w_{ij} = \eta x_j(t+1) y_i = \eta (x_j(t) - \sum_k w_{kj} y_k) y_i \tag{4}$$

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

and so this network not only causes convergence of the weights but causes the weights to converge to span the subspace of the Principal Components of the input data.

However the PCA network cannot be used to identify individual bars[1] and so we subsequently have shown that Nonlinear PCA (NLPCA) in which (1) is substituted by

$$y_i = f(\sum_{j=1}^{N} w_{ij} x_j) \qquad (5)$$

can, with suitable choice for the nonlinearity, f(), identify the individual bars.
In summary, our results were that

- With less than 16 output neurons, all bars are discovered but some outputs will have found two or more causes simultaneously.
- If the network has 16 neurons then all bars are discovered individually.
- With more than 16 neurons, all sources are found but in some cases two or more outputs will have discovered the same bar.

3. Hierarchical Feature Extraction

We now investigate a data set which was used in [6] to illustrate composite structure on this simple bars data set: with probability of 0.6, the bars data[1] set described above is used with each individual bar appearing with probability of 0.3. However, with probability 0.4, one of the composite structures shown in Figure 3 is selected, each composite structure having an equal probability of 0.1 of being chosen.

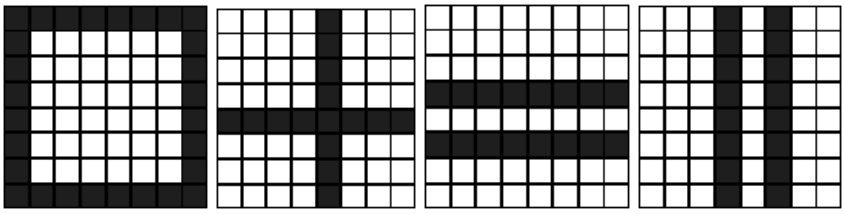


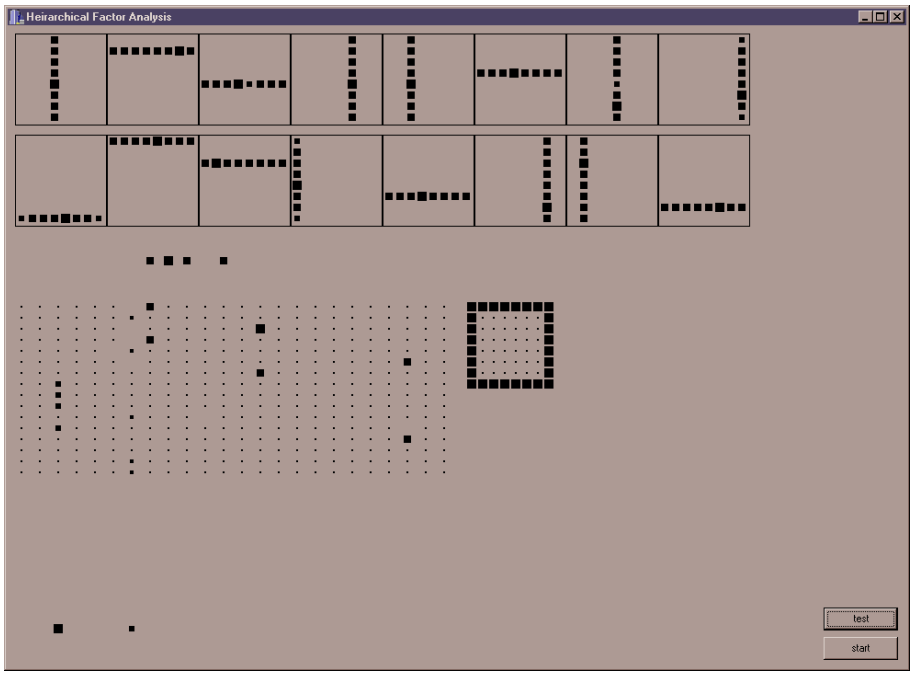
Figure 3: The composite bars data used in hierarchical feature extraction.

The aim of the simulation is firstly to identify all underlying causes of the data set – the individual bars - and secondly to find the composite features shown in Figure 3 which will appear more often than randomly. We have a two layer neural network in which both layers use NLPCA with identical functions. The first layer (Figure 4) successfully identifies all of the individual bars while the second layer extracts the composite structure. The top half of Figure 4 represents the weights from the input layer to the middle layer where each weight vector has been arranged to correspond to an input square; a black square indicates a large weight from the input square to

[1] Actually, a 5*5 grid was used in [5]

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

that middle layer neuron. The bottom half of the Figure shows the weights from the



middle layer to the output layer (we have used 24 output neurons so as not to prejudge the number of composite features, see later). The third output neuron is firing most strongly (last line of Figure). Its weights show that it is linked most strongly to the 8^{th} , 9^{th} , 10^{th} and 12^{th} hidden neurons which correspond to the input bars from which the pattern was formed.

Figure 4: The top line shows the weights of the converged network from input layer to second layer. These weights have clearly identified the individual bars. The outputs of this second layer are shown underneath this for the square input pattern shown. The bottom half of the figure shows the weights from second to third layer. The four multiple causes have been identified by different individual neurons and the other neurons do not respond to the input pattern.

From the weights into the third layer, we see that the other 4 composite patterns have been identified by output neurons 8, 14 and 22. Notice that the other output neurons have not shown any response to any hidden neuron firing whereas in our previous work[1] each symbol would be shared by several output neurons. This is due to added noise on the output neurons before feed back.

Non-linear PCA can be shown to be an approximation to the minimisation[8] of $\mathbf{J} = \mathbf{E}(\mathbf{x} - \mathbf{Xy})^2 = \mathbf{E}(\mathbf{x} - \mathbf{Wf}(\mathbf{a}))^2$ where $\mathbf{E}()$ is the expectation operator. Now

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

we add noise to the process so that $\mathbf{y} = \mathbf{f}(\mathbf{a}) + \boldsymbol{\mu}$ where μ is a vector os independantly drawn noise from a zero mean distribution. So defining $\mathbf{f} = \mathbf{f}(\mathbf{a})$,

$$\begin{aligned}
\mathbf{J'} &= \mathbf{E}(\mathbf{x} - \mathbf{W}\mathbf{y})^2 \\
&= \mathbf{E}(\mathbf{x} - \mathbf{W}(\mathbf{f} + \boldsymbol{\mu}))^2 \\
&= \mathbf{E}\left\{\mathbf{x}\mathbf{x}^{\mathbf{T}} - \mathbf{x}(\mathbf{f} + \boldsymbol{\mu})^{\mathbf{T}}\mathbf{W}^{\mathbf{T}} - \mathbf{W}(\mathbf{f} + \boldsymbol{\mu})\mathbf{x}^{\mathbf{T}} + \mathbf{W}(\mathbf{f} + \boldsymbol{\mu})(\mathbf{f} + \boldsymbol{\mu})^{\mathbf{T}}\mathbf{W}^{\mathbf{T}}\right\} \\
&= \mathbf{E}\left\{\mathbf{x}\mathbf{x}^{\mathbf{T}} - \mathbf{x}\mathbf{f}^{\mathbf{T}}\mathbf{W}^{\mathbf{T}} - \mathbf{W}\mathbf{f}\mathbf{x}^{\mathbf{T}} + \mathbf{W}\mathbf{f}\mathbf{f}^{\mathbf{T}}\mathbf{W}^{\mathbf{T}} + \mathbf{W}\boldsymbol{\mu}\boldsymbol{\mu}^{\mathbf{T}}\mathbf{W}^{\mathbf{T}}\right\} \\
&= \mathbf{J} + \mathbf{W}\mathbf{D}\mathbf{W}^{\mathbf{T}}
\end{aligned} \tag{6}$$

Where **D** is a diagonal matrix with positive elements on the diagonal equal to the variance of the noise. This is very like a constrained optimisation problem where the D matrix represents a set of Lagrange multipliers. Now if J' were to be produced as a constrained maximisation of J under the constraint that $WW^T > 0$, the Kuhn-Tucker equations show that when the Lagrange multipliers are equal to 0, the maximum is the maximum of J while for multipliers ≠ 0, the weights, W should be equal to zero. Thus we are forcing weights not involved in the minimisation to 0.

Intuitively, we can see that, with a low magnitude of added noise to the network the first term of (6) dominates and so non-linear PCA is performed in the normal manner. If the noise level is increased then the learning is moderated by this additional weighted noise term which has the effect of forcing some weight vectors to zero (the degenerate solution) in order to maintain the condition that the weight vectors must be orthogonal. Extracting a weight update rule from the energy equation (4), we have $\Delta\mathbf{W} \propto \mathbf{f}(\mathbf{a})(\mathbf{x} - \mathbf{W}\mathbf{f}(\mathbf{a})) - \mathbf{W}\mathbf{D}$ As the amplitude of the noise becomes larger then the right term has more effect and pushes the weight values downwards. So the introduction of noise onto the outputs has the effect of introducing a second pressure into the learning rule of the non-linear PCA algorithm.This is a natural way in which to introduce a sparsification term onto the weights which is similar to the weights penalty term described by [9].

The non-linearity used in this experiment is the threshold log function (also used in [7]),

$$y_{(t+1)} = \alpha * \log\left(1 + \frac{\exp(y_t - \vartheta)}{\sigma}\right)$$

The layers of the network were trained separately: the first layer was trained for 100,000 epochs and then its weights were set while activation was passed from inputs to hidden neurons to output neurons and the second set of weights was trained. When data that was not one of the four higher order structures is used as input(e.g. any random combination of the bars) the second layer of the network does not respond at all. However when presented with results containing higher order

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

structure plus extra bars the network continues to identify this structure (Figure 5). We see that the middle layer is identifying all of the individual factors (the bars) in the data set while the last layer is identifying the composite factors in the data set.

4. Face Identification

We now use the above network for face identification; our data set comprises images of faces of 6 subjects in each of 5 non-standard poses. By non-standard, we mean that we do not use faces which are always in the same 5 poses for each of the 6 subjects. We recall that the network is designed to give a sparse coding of the data set; however such a coding is only possible where the data set itself is amenable to sparsification. We therefore preprocess our data set by edge filtering. It is our experience that applying the hierarchical Nonlinear PCA network on grey scale images without this preprocessing does not give accurate sparse coding of the data set. Outputs from the first layer of weights are shown in Figure 6: the first 5 lines show the coding of the network to the first subject in each of 5 poses, lines 6-10 show the coding of the network to the 5 different images of subject 2 etc.

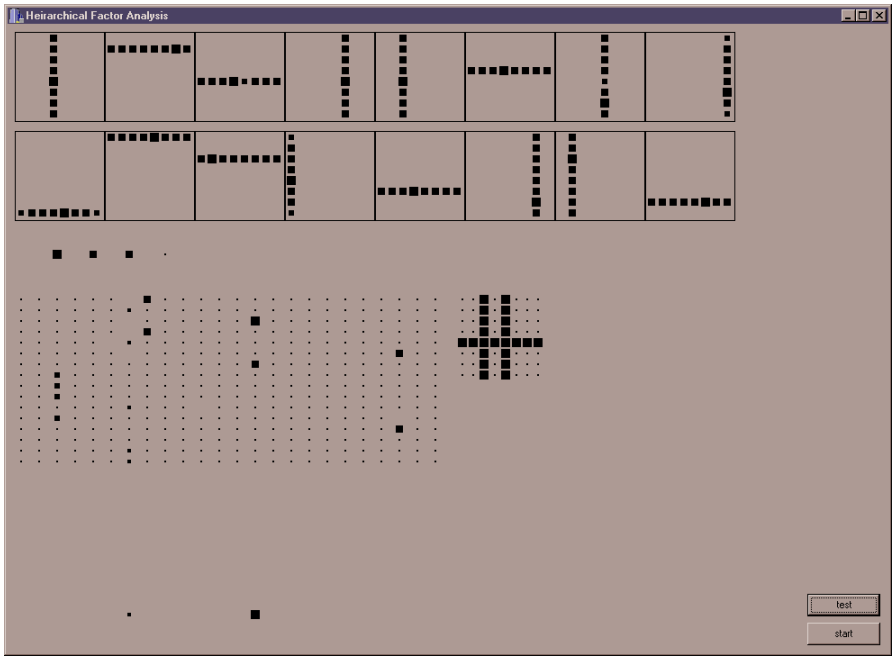


Figure 5: The higher order structure is masked by an extra bar. The network still correctly identifies the structure.

ESANN'1999 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 21-23 April 1999, D-Facto public., ISBN 2-600049-9-X, pp. 281-288

Best results were achieved with $f(y) = \frac{1}{(1+\alpha\exp(\sigma-\theta y_t))}$ as our nonlinear function, where θ =0.1 and σ =4. This is similar in shape to the previous non-linearity but is more effective at forcing outputs to 1 and 0 (left half of Figure 6). The coding has a mode of 5 outputs on (white = 1) of the 10 output neurons though coding is as sparse as 1 of 10 firing. Note the binary nature of the coding: white rectangles represent weights >0.95 while black rectangles represent weights < 0.05

5. Conclusion

We have extended our previous work on using nonlinear PCA to identify factors in a data set by using a two layer model which has been shown to be capable of identifying composite features consisting of higher order structure in the data set. We have illustrated our method on artificial and real data. Further work will investigate the scaling properties of the network and whether the coding can be used to reliably interpolate between faces seen in training.

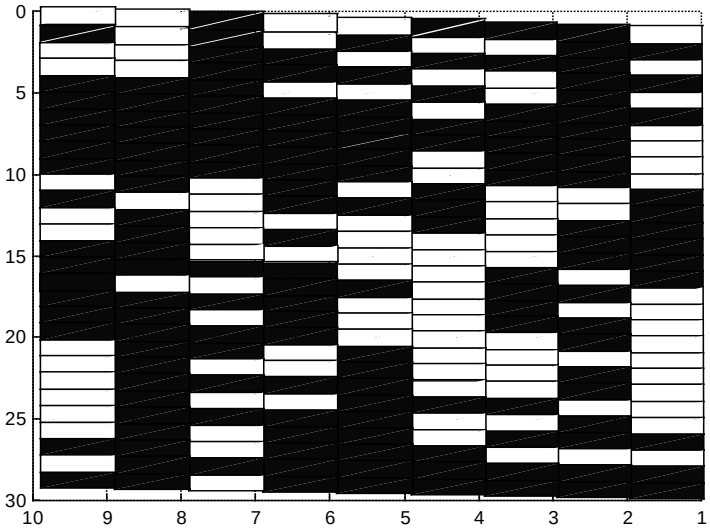


Figure 6: The first level outputs when the data set consists of 6 faces in each of 5 different poses. The images are arranged so that lines 1-5 are the coding of the first person, lines 6-10 are the coding of the second etc.

References

[1] Charles, D. and Fyfe, C. "Modelling Multiple Cause Structure using Rectification Constraints", Network: Computation in Neural Systems, 9:167-182,1998.

[2] Fyfe, C. “PCA properties of interneurons”, From Neurobiology to Real World Computing, proceedings of International Conference on Artificial Neural Networks, 1993.

[3] Barlow H B, “Unsupervised Learning” Neural Comp. 1 295-311 1989.

[4] Foldiak P., “Models of sensory coding” Ph.D. thesis, University of Cambridge, 1992.

[5] Lewicki, M. S. and Sejnowski, T. J. “Bayesian Unsupervised Learning of Higher Order Structure”, NIPS 9 529-534, 1996.

[6] Webber C, “Generalisation and discrimination emerge from a self-organising componential network: a speech example”, Network: Computation in Neural Systems,8:425-439,1997.

[7] Karhunen, J and Joutsensalo, J, “Representation and separation of signals using non-linear PCA type learning”, Neural Networks, 7(1):113-127,1994

[8] Harpur G and Pragur, R, ”Development of entropy coding in a recurrent network”, Network 7:277-284,1996