ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

Maximum Likelihood Hebbian Rules

Colin Fyfe and Emilio Corchado

Applied Computational Intelligence Research Unit
The University of Paisley
Scotland
colin.fyfe, emilio.corchado@paisley.ac.uk

Abstract: In this paper, we review an extension of the learning rules in a Principal Component Analysis network which has been derived to be optimal for a specific probability density function. We note that this probability density function is one of a family of pdfs and investigate the learning rules formed in order to be optimal for several members of this family. We show that, whereas previous authors [5] have viewed the single member of the family as an extension of PCA, it is more appropriate to view the whole family of learning rules as methods of performing Exploratory Projection Pursuit. We illustrate this on artificial data sets.

1. Introduction

Principal Component Analysis (PCA) is a standard statistical technique for compressing data; it can be shown to give the best linear compression of the data in terms of least mean square error. There are several artificial neural networks which have been shown to perform PCA e.g. [8, 9]. We shall be most interested in a negative feedback implementation [3].
The basic PCA network [3] 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*. η is a learning rate. Then the activation passing and learning is described by

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

$$e_j = x_j - \sum_{i=1}^{M} W_{ij} y_i \tag{2}$$

$$\Delta W_{ij} = \eta e_j y_i \tag{3}$$

The weights converge to the Principal Component directions.

Exploratory Projection Pursuit (EPP) is a more recent statistical method aimed at solving the difficult problem of identifying structure in high dimensional data. It does this by projecting the data onto a low dimensional subspace in which we search for its structure by eye. However not all projections will reveal the data's structure equally well. We therefore define an index that measures how “interesting" a given projection is, and then represent the data in terms of projections that maximise that index. Now

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

"interesting" structure is usually defined with respect to the fact that most projections of high-dimensional data onto arbitrary lines through most multi-dimensional data give almost Gaussian distributions [2]. Therefore if we wish to identify "interesting" features in data, we should look for those directions onto which the data-projections are as far from the Gaussian as possible.
In this paper, we derive a neural method of performing Exploratory Projection Pursuit from a probabilistic perspective.

2. Maximum Likelihood Hebbian Learning

It has been shown [11] that the learning rule

$$\Delta W_{ij} = \eta\left(x_j y_i - y_i \sum_k W_{kj} y_k \right) \tag{4}$$

can be derived as an approximation to the best linear compression of the data.
Thus we may start with a cost function

$$J(W) = 1^T E\left\{(\mathbf{x} - W\mathbf{y})^2\right\} \tag{5}$$

which we minimise to get the rule(4). [5] used the residual in (5) to define a cost function of the residual

$$J = f_1(\mathbf{e}) = f_1(\mathbf{x} - W\mathbf{y}) \tag{6}$$

where $f_1 = \|.\|^2$ is the (squared) Euclidean norm in the standard PCA rule.
We may show [1] that the minimization of J is equivalent to minimizing the negative log probability of the residual, $\mathbf{e}$, if $\mathbf{e}$ is Gaussian.

Let $p(\mathbf{e}) = \frac{1}{Z}\exp(-\mathbf{e}^2)$ (7)

Then we can denote a general cost function associated with this network as

$$J = -\log p(\mathbf{e}) = (\mathbf{e})^2 + K \tag{8}$$

where K is a constant. Therefore performing gradient descent on J we have

$$\Delta W \propto -\frac{\partial J}{\partial W} = -\frac{\partial J}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial W} \approx \mathbf{y}(2\mathbf{e})^T \tag{9}$$

where we have discarded a less important term (see [7] for details).In general [10], the minimisation of such a cost function may be thought to make the probability of the residuals greater dependent on the pdf of the residuals. Thus if the probability density function of the residuals is known, this knowledge can be used to determine the optimal cost function which in turn gives an optimal learning rule. This suggests a family of learning rules which are derived from the family of exponential distributions. Let the residual after feedback have probability density function

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

$$p(\mathbf{e}) = \frac{1}{Z}\exp(-|\mathbf{e}|^p)\,. \tag{10}$$

Then we can denote a general cost function associated with this network as

$$J = -\log p(\mathbf{e}) = |\mathbf{e}|^p + K \tag{11}$$

where K is a constant. Therefore performing gradient descent on *J* we have

$$\Delta W \propto -\frac{\partial J}{\partial W} = -\frac{\partial J}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial W} \approx y(p\,|\mathbf{e}|^{p-1}\ sign(\mathbf{e}))^T \tag{12}$$

where T denotes the transpose of a vector. We would expect that for leptokurtotic residuals (more kurtotic than a Gaussian distribution), values of p<2 would be appropriate, while for platykurtotic residuals (less kurtotic than a Gaussian), values of p>2 would be appropriate. It is a common belief in the ICA community [6] that it is less important to get exactly the correct distribution when searching for a specific source than it is to get an approximately correct distribution i.e. all supergaussian signals can be retrieved using a generic leptokurtotic distribution and all subgaussian signals can be retrieved using a generic platykurtotic distribution. Our experiments will tend to support this belief to some extent but we often find accuracy and speed of convergence are improved when we are accurate in our choice of p.
Therefore the network operation is as before except:

Weight change: $$\Delta W_{ij} = \eta.y_i.sign(e_j)|\,e_j\,|^{p-1} \tag{13}$$

[5] described their rule as performing a type of PCA, but this is not strictly true since only the original Hebbian rule (3) actually performs PCA. By maximising the likelihood of the residual with respect to the actual distribution, we are matching the learning rule to the pdf of the residual. We may thus link the method to the standard statistical method of Exploratory Projection Pursuit. Now the nature and quantification of the interestingness is in terms of how likely the residuals are under a particular model of the pdf of the residuals.

3. Experimental Results

To illustrate our method, we follow [4] in creating artificial data sets, each of 10 dimensions. All results reported are based on a set of 10 simulations each with different initial conditions. It is our general finding that sphering is necessary to get the most accurate results presented below.
In the first data set, we have 9 leptokurtotic dimensions and one gaussian dimension; this is almost the opposite of the standard EPP data sets described in [4] and is rather far from being a typical data set in that most of the directions in terms of its natural basis are non-Gaussian. However, since we wish to investigate our new models, it is a good test set since we can easily see the results of our method. We wish to identify the single Gaussian dimension and ignore the leptokurtotic dimensions. The leptokurtotic dimensions may be characterised as having long tails; if a residual can be created by removing the Gaussian direction from the data set, the residual will automatically be leptokurtotic. Thus we consider maximising the likelihood of the residual using the model

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

$$p(\mathbf{e}) = \frac{1}{Z}\exp(-|\mathbf{e}|^p) \text{ with p<2;} \quad (14)$$

We have experimented with a number of values of p and report on simulations with p=1.5. A typical result is shown in Figure 1; the Gaussian direction is clearly identified.

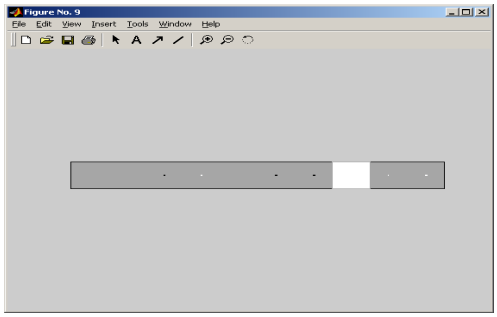

Figure 1: The Gaussian direction was the third among 9 leptokurtotic dimensions. It has clearly been identified in this Hinton map of the weights.

We have similar results with a data set containing 9 platykurtotic dimensions and one Gaussian dimension. We use the same learning rules as before but with a value of p=3. If our data set consists of 9 Gaussian dimensions and 1 leptokurtotic dimension, we can identify the leptokurtotic dimension with a rule using p>2. This is really saying that all residuals will be unlikely using this model but that the leptokurtotic dimension is more wrong under the platykurtotic model than the Gaussian dimensions and should be removed from the residual. In the next section, we derive an alternative method for this data set.

4. Minimum Likelihood Hebbian Learning

Just as the Hebbian learning rule has an opposite known as the anti-hebbian rule, we may change our rules so that

$$\Delta W \propto \frac{\partial J}{\partial W} = \frac{\partial J}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial W} \approx -\mathbf{y}(p\,|\mathbf{e}|^{p-1}\ sign(\mathbf{e}))^T \quad (15)$$

Now we may argue that, in doing so, we are aiming to minimise the likelihood of the residual given the current model. In detail, if the residual has probability density function

$$p(\mathbf{e}) = \frac{1}{Z}\exp(-|\mathbf{e}|^p)\,. \quad (16)$$

and we denote the general cost function associated with this network as

$$J = -\log p(\mathbf{e}) = |\mathbf{e}|^p + K \quad (17)$$

where K is a constant, we may perform gradient ascent on J to get

$$\Delta W \propto \frac{\partial J}{\partial W} = \frac{\partial J}{\partial \mathbf{e}}\frac{\partial \mathbf{e}}{\partial W} \approx -y(p\,|\mathbf{e}|^{p-1}\ sign(\mathbf{e}))^T \quad (18)$$

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

We are thus using our learning rules to make the residuals as unlikely as possible under the current model assumptions (determined by the *p* parameter). Thus when we have 9 Gaussian dimensions and 1 platykurtotic dimension we get results as in Figure 2 (with p=3 in our minimum likelihood rule). By identifying and removing the platykurtotic dimension we are leaving a residual which has 0 kurtosis.

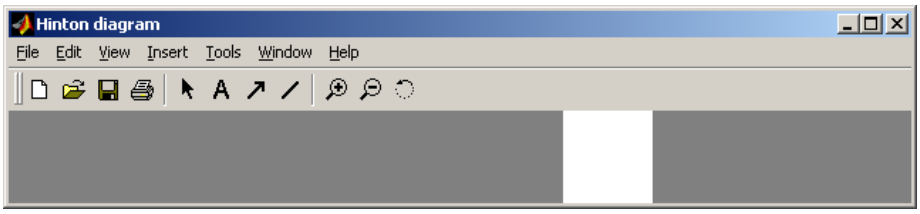

Figure 2: The platykurtotic dimension has been identified among the gaussian dimensions.

Note that with Minimum Likelihood Hebbian learning we are using the correct model for the distribution that we are seeking but minimising the probability of the residual being taken from this distribution. Thus we find and extract this distribution.

5. Discussion

In this paper, we have restricted our learning rules to those drawn from the exponential family of distributions. All of the artificial data sets above also came from this family of distributions and we might legitimately ask whether these rules will work on data sets which are not drawn from this family. For example, the last data set was slightly changed to 9 Gaussian dimensions and one drawn from the Beta(2,2) distribution. We chose the Beta distribution since it is very malleable and we chose these parameters since it is then not unlike a Gaussian in shape.
Using p=3 in our family of rules we consistently found the beta distribution. We might go on to ask whether the beta distribution has to be mixed with Gaussian distributions and so we create a similar data set with 9 platykurtotic exponential dimensions and one beta function dimension, •(0.5, 0,5). We have used Minimum Likelihood learning and p=3 to find the • distribution. Since the • function has a non zero mean, this mean has been subtracted from the data.
We used these values of the • parameters since the difference in kurtosis between the platykurtotic dimensions and the beta dimension is very small, ~ 0.1 (see Table 1)

Dim	1	2	*3*	4	5	6	7	8	9	10
Kurt	1.37	1.38	***1.49***	1.38	1.38	1.37	1.38	1.39	1.38	1.38

Table 1. The single • dimension (Dimension 3) has slightly more kurtosis than the others.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 143-148

In conclusion, we have derived a family of learning rules based on the probability density function of the residuals. This family of rules may be called Hebbian in that all use a simple multiplication of the output of the neural network with some function of the residuals after feedback. The power of the method comes from the choice of an appropriate function. In particular, we showed how to choose a function to maximise the likelihood of the residuals under particular models of probability density functions. We now see that both the original PCA rule and the •-insensitive rule [5] are merely particular cases of this class of rules. We have also shown that the rules are more akin to Exploratory Projection Pursuit and prefer to call them Maximum Likelihood Hebbian learning, believing that '•-insensitive PCA' does not do justice to the power of the method. We have also shown how powerful Minimum Likelihood Hebbian learning is and indeed that this is, in some sense, even more closely related to EPP: the real power of these learning rules is in the context of exploratory data analysis. These are powerful new tools for the data mining community and should take their place along with existing exploratory methods.

References

[1] Bishop, C.M, Neural Networks for Pattern Recognition, Oxford, 1995.
[2] Diaconis, P. and Freedman D., Asymptotics of Graphical Projections. The Annals of Statistics. 12(3): 793-815, 1984.
[3] Fyfe, C., "PCA Properties of Interneurons", From Neurobiology to Real World Computing, Proceedings of International Conference on Artificial on Artificial Neural Networks, ICAAN 93, pages 183-188, 1993.
[4] Fyfe, C. and Baddeley, R. Non-linear Data Structure Extraction using Simple Hebbian Learning, Biological Cybernetics,72(6), 533-541,1995.
[5] Fyfe, C. and MacDonald, D., ε-Insensitive Hebbian learning, Neurocomputing, 2001.
[6] Hyverinnen, A. Complexity Pursuit: Separating interesting components from time series. Neural Computation, 13: 883-898, 2001.
[7] Karhunen, J. and Joutsensalo, J., Representation and Separation of Signals Using Non-linear PCA Type Learning, Neural Networks, 7:113-127, 1994.
[8] Oja, E., Neural Networks, Principal Components and Subspaces, International Journal of Neural Systems, 1:61-68, 1989.
[9] Oja, E., Ogawa, H., Wangviwattana, J., Principal Components Analysis by Homogeneous Neural Networks, part 1, The Weighted Subspace Criterion, IEICE Transaction on Information and Systems, E75D: 366-375,May 1992.
[10] Simola, A.J. and. Scholkopf, B. A Tutorial on Support Vector Regression. Technical Report NC2-TR-1998-030, NeuroCOLT2 Technical Report Series, Oct.1998.
[11] Xu L., Least Mean Square Error Reconstruction for Self-Organizing Nets", Neural Networks, Vol. 6, pp. 627-648, 1993.