ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

Probabilistic Derivation and Multiple Canonical Correlation Analysis

Pei Ling Lai
Department of Computer Science,

York University,
UK.
email: pllai@cs.york.ac.uk

Abstract. We review a new method of performing Canonical Correlation Analysis (CCA) with Artificial Neural Networks. We have previously [5, 4] compared its capabilities with standard statistical methods on simple data sets where the maximum correlations are given by linear filters. In this paper, we extend the method by implementing a very precise set of constraints which allow multiple correlations to be found at once. We demonstrate the network's capabilities on the standard random dot stereogram data set. We also re-derive the learning rules from a probabilistic perspective and then by use of a specific prior on the weights, simplify the algorithm. We demonstrate its capabilities on a standard problem [2]which is an abstraction of the Random Dot Stereogram matching problem and show how a second layer network using Factor Analysis can be used to combine the results of the CCA network to obtain higher order information.

1 Introduction

Canonical Correlation Analysis(CCA) is a statistical technique used when we have two data sets which we believe have some underlying correlation. Consider two sets of input data; $\mathbf{x}_1$ and $\mathbf{x}_2$. Then in classical CCA, we attempt to find that linear combination of the variables which gave us maximum correlation between the combinations. Let

$$y_1 = \mathbf{w}_1\mathbf{x}_1 = \Sigma_j w_{1j}x_{1j} \qquad \text{and} \qquad y_2 = \mathbf{w}_2\mathbf{x}_2 = \Sigma_j w_{2j}x_{2j}$$

where we have used w_{2j} as the j^{th} element of the parameter vector $\mathbf{w}_2$ etc. Then we wish to find those values of $\mathbf{w}_1$ and $\mathbf{w}_2$ which maximize the correlation between y_1 and y_2. Whereas Principal Components Analysis, Factor Analysis or Exploratory Projection Pursuit deal with the inter-relationships within a set of variables, CCA deals with the relationship between two sets of variables.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

In this paper, we review a neural method of finding canonical correlations in Section 2 and extend the method to find multiple correlation simultaneously even where these correlations have the same magnitude in Section 3. In Section 4, we rederive from a probabilistic perspective a neural implementation of CCA which adaptively learns the optimal weights to maximize correlations between the data sets and then use the new derivation to derive a nonlinear extension of CCA. We then illustrate the method on the abstraction of the Random Dot Stereogram matching problem popularized by Becker [2].

2 The Canonical Correlation Analysis Network

We wish to maximize the correlation $E(y_1 y_2)$ where $E()$ denotes the expectation which will be taken over the joint distribution of $\mathbf{x}_1$ and $\mathbf{x}_2$. We may regard this problem as that of maximizing the function $g_1(\mathbf{w}_1|\mathbf{w}_2) = E(y_1 y_2)$ which is defined to be a function of the weights, $\mathbf{w}_1$ given the other set of parameters, $\mathbf{w}_2$. This is an unconstrained maximization problem which has no finite solution and so we must constrain the maximization. Typically in CCA, we add the constraint the $E(y_1^2 = 1)$ and similarly with y_2^2 when we maximize $g_2(\mathbf{w}_2|\mathbf{w}_1)$. Using the method of Lagrange multipliers, this yields the constrained optimization function:

$$J = E\{(y_1 y_2) + \frac{1}{2}\lambda_1(1 - y_1^2) + \frac{1}{2}\lambda_2(1 - y_2^2)\} \quad (1)$$

We use gradient ascent on the instantaneous version of this function with respect to both $\mathbf{w}_1$ and $\mathbf{w}_2$ and the Lagrange multipliers, λ_1 and λ_2. By changing the Lagrange multipliers in proportion to the derivates of J we are changing the relative strength of the constraint compared to the function we are optimizing; this allows to smoothly maximize that function in the region in which we are satisfying the constraint. Noting that

$$\frac{\partial g_1(\mathbf{w}_1|\mathbf{w}_2)}{\partial \mathbf{w}_1} = \frac{\partial (y_1 y_2)}{\partial \mathbf{w}_1} = \frac{\partial (\mathbf{w}_1 \mathbf{x}_1 y_2)}{\partial \mathbf{w}_1} = \mathbf{x}_1 y_2$$

these yield respectively

$$\frac{\partial J_1}{\partial \mathbf{w}_1} = \mathbf{x}_1 y_2 - \lambda_1 y_1 \mathbf{x}_1 = \mathbf{x}_1(y_2 - \lambda_1 y_1)$$

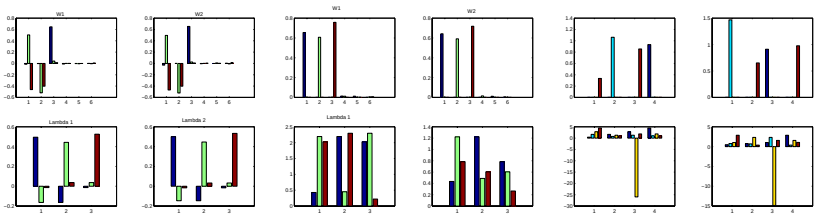
$$\frac{\partial J_1}{\partial \lambda_1} \propto (1 - y_1^2)$$

We therefore have the joint learning rules

$$\Delta w_1 = \eta x_1(y_2 - \lambda_1 y_1) \qquad \lambda_1 = \eta_0(1 - y_1^2)$$
$$\Delta w_2 = \eta x_2(y_1 - \lambda_2 y_2) \qquad \lambda_2 = \eta_0(1 - y_2^2)$$

We have previously [5] illustrated this networks's capabilities on both real and artificial data sets.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452



(a) (b) (c)

Figure 1: (a)The top row shows the weights of W_1 and W_2. (The first column in each of the six blocks is the weight into the first output neuron etc. The weights span the three dimensional subspace of the first three correlations. The bottom row shows the Λ_1 and Λ_2 matrices which are almost diagonal. (b)The top row shows the weights of W_1 and W_2. They find the actual first three correlations. The bottom row shows the Λ_1 and Λ_2 matrices. (c)The top pair of diagrams show $\mathbf{w}_1$ and $\mathbf{w}_2$ which show that 1 pair have learned the right shift while two other pairs have learned the left shift. The bottom diagrams show the Λ matrices which explains why the fourth pair of neurons is inactive.

3 Many Correlations

However the preceding method only enables us to find one correlation in a data set. We can find more by deflationary methods, however a better method is to create an objective function which contains the necessary criteria for finding more than one correlation. One obvious basis for this would be to insist that $\mathbf{y}_i\mathbf{y}_i^T = I$ where $\mathbf{I}$ is the m*m identity matrix, with $\mathbf{y}_i, i = 1, 2$ the vector of first (resp. second) outputs.

Thus the criterion becomes maximize

$$J \quad = \quad E\{(\mathbf{y}_1^T\mathbf{y}_2) + \lambda_1(\mathbf{y}_1\mathbf{y}_1^T - I) + \lambda_2(\mathbf{y}_2\mathbf{y}_2^T - I)\}$$

with $\lambda_i, i = 1, 2$ now a matrix of Lagrange multipliers. This give us the learning rules

$$\Delta W_1 \quad = \quad \eta\mathbf{x}_1(\mathbf{y}_2 - \lambda_1\mathbf{y}_1)^T \qquad \Delta\lambda_1 = \eta_0(\mathbf{y}_1\mathbf{y}_1^T - I) \quad \text{where} \qquad \mathbf{y}_1 = W_1^T\mathbf{x}_1$$

with W_1 the m*n matrix of weights connecting $\mathbf{x}_1$ to $\mathbf{y}_1$. Similarly with W_2, λ_2. With this formulation we find a rotation of the canonical correlation. e.g. with artificial data in which $\mathbf{x}_1$ and $\mathbf{x}_2$ are both 6 dimensional data vectors each of whose elements are randomly drawn from the zero mean Gaussian distribution $N(0, 1)$ before an additional 3 samples are drawn from the same distribution and added to the first three elements of $\mathbf{x}_1$ and $\mathbf{x}_2$, we find the results in Figure 1a. The fact that there are correlation in the first three elements has clearly been found and we see that there is only a very small magnitude in the other three positions of W_1 and W_2.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

However the above network only finds the correlations spread out over the subspace spanned by the major correlations. We are constraining the expected value of $\mathbf{y}_1\mathbf{y}_1^T = I$ which means that

$$E(W_1^T\mathbf{x}_1\mathbf{x}_1^T W_1) \quad = \quad I \quad \text{ i.e. } \quad W_1^T\Sigma_1 W_1 = I \quad \text{ or } \quad W_1^T W_1 = I \text{ if } \Sigma_1 = I$$

where we have used Σ_1 as the covariance matrix of the input vector $\mathbf{x}_1$. Now for our artificial data set this last condition holds and so any orthogonal matrix satisfies the constraints. The λ_1 matrix learns the diagonal elements to ensure the $E(\mathbf{y}_{1i}^2) = 1$ but there is no interaction between the elements in $\mathbf{x}_1$ so that we do not ensure that $E(\mathbf{y}_{1i}\mathbf{y}_{1j}) = 0, i \neq j$ since $(\Lambda_1)_{ij} = 0$. We have chosen in Figure 1b an example of the Λ matrix in which the off-diagonal elements have grown somewhat larger than usual (though still very much smaller than the diagonal elements) to illustrate the small interaction which there is between the first two correlations.

A recent trick [3] to turn a PCA network into a Factor Analysis network has been to insist that the weights remain positive. Recognizing that the correlations we wish to find are positive, we may insist that the weights remain positive i.e. should the weight change mean that the weights become negative, we simply set that weight to 0. It subsequently has the opportunity to grow positive again. This is not too restrictive a practice since we may insist that another output neuron has negative weights if we suspect an anticorrelation between $\mathbf{x}_1$ and $\mathbf{x}_2$. This change is also motivated by Dale's law which states that a neuron may be inhibitory or excitatory but may not switch from one effect to the other. With this additional constraint we have the results shown in Figure 1b: the actual correlation filers have been found and we see that the off-diagonal elements of $\lambda_i, i = 1, 2$ are very much larger.

3.1 Random Dot Stereograms

With our previous single neuron constraint, we required to enforce a competition between pairs of outputs (one in $\mathbf{y}_1$ and one in $\mathbf{y}_2$) to ensure that one pair learned the left shift while another learned the right shift. This added additional complexity to the model. With the above network such a competition is not necessary: we see in Figure 1c that two pairs of neurons have learned a left shift while one has learned a right shift. This experiment was done with the $\mathbf{y}_i$ vectors being 4-dimensional. We see that one pair of output neurons' weights have not grown and the reason for this is seen in the large magnitude of the Lagrange values for that output.

4 A Probabilistic Perspective

We may also derive these learning rules from a probabilistic perspective. Let us assume that we have discovered the best linear combination of the elements of $\mathbf{x}_2$ which will match with greatest correlation a linear combination of $\mathbf{x}_1$. Then y_2 has the target distribution which we wish y_1 to match as closely as

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

possible. Now if the distributions match exactly, there is a linear relationship between y_1 and y_2 i.e.

$$y_2 \quad = \quad \lambda_1 y_1 + \theta$$

For simplicity in the following, we will take $\theta = 0$. Now let us assume that the actual value of y_2 is gaussian distributed with mean proportional to y_1. Then the probability of the output y_2 given the model and its parameters is

$$P(y_2|\mathbf{w}_1, \mathbf{x}_1, H) \quad = \quad \frac{1}{Z_n} \exp\{-\beta(\lambda_1 y_1 - y_2)^2)\}$$

where Z_n is a normalising factor, H is the current model and β determines the spread of the distribution. We wish to find the most probable value of y_2. Then we may define J_1 to be the negative log of this probability to give an objective function which we can minimise with respect to the parameters $\mathbf{w}_1$ to get the learning rules

$$\Delta \mathbf{w}_1 \propto -\frac{\partial J_1}{\partial \mathbf{w}_1} = \beta \mathbf{x}_1 (y_2 - \lambda_1 y_1)\lambda_1 \tag{2}$$

The learning rule for $\mathbf{w}_1$ is identical to that above with $\beta\lambda_1 = \eta$. This perspective also allows us to perform error descent on the unknown parameter λ_1 (ignoring for the moment difficulties with joint optimisations) or we may set λ_1 to a particular value determined by our prior beliefs about the data set, $\mathbf{x}_1$ and $\mathbf{x}_2$ and the parameters. Now we may consider the more general situation where

$$y_1 \quad = \quad f_1(\mathbf{w}_1 \mathbf{x}_1)$$

for some nonlinear function $f_1()$. Then

$$\Delta \mathbf{w}_1 = -\frac{\partial J_1}{\partial \mathbf{w}_1} = \beta f_1' \mathbf{x}_1 (y_2 - \lambda_1 y_1)\lambda_1 \tag{3}$$

where f_1' is the derivative of $f_1(t)$ with respect to t.

Let us now consider our prior beliefs about the weights $\mathbf{w}_1$. It is our prior belief that we require a non-zero value for y_1; in other words, we wish to ensure that our learning algorithm does not find values of $\mathbf{w}_1$ and $\mathbf{w}_2$ that give us the trivial solution $y_1 = y_2 = 0$. For reasons which will become obvious later, we will use the prior

$$P(\mathbf{w}_1|\mathbf{x}_1, H) = \frac{1}{Z_w} \exp(\frac{1}{2}\gamma y_1^2) \tag{4}$$

where Z_w is a normalising constant. We may equally express this as a prior on y_1 since the relationship between $\mathbf{w}_1$ and y_1 is deterministic. Note that this implies that our prior belief in the joint magnitude of all weights into output y_1 is an exponential distribution. Now $\frac{1}{Z_w} \exp(\frac{1}{2}\gamma y_1^2) = \frac{1}{Z_w} \exp(\frac{1}{2}\gamma f_1^2(\mathbf{w}_1 \mathbf{x}_1)))$ and so with binary $\mathbf{x}_1$, we are at or close to the corners of a hypercube. Since

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

most of the volume of a high dimensional hypercube is near the corners, this prior displays our belief that some weights will have large values. Then the joint probability of weights and data can be written

$$P(y_2, \mathbf{w}_1|\mathbf{x}_1, H) = P(y_2|\mathbf{w}_1, \mathbf{x}_1, H).P(\mathbf{w}_1|\mathbf{x}_1, H)$$

and so taking

$$J_2 = -\log(P(y_2|\mathbf{w}_1, \mathbf{x}_1, H).P(\mathbf{w}_1|\mathbf{x}_1, H))$$

we have

$$\Delta\mathbf{w}_1 \quad = \quad y_2\mathbf{x}_1\beta\lambda_1 f_1' - \mathbf{x}_1 y_1(\beta\lambda_1^2 - \gamma)f_1'$$

If we use $y_1 = \tanh(\mathbf{w}_1\mathbf{x}_1)$

$$\Delta\mathbf{w}_1 = y_2\mathbf{x}_1\beta\lambda_1(1 - y_1^2) - \mathbf{x}_1 y_1(1 - y_1^2)(\beta\lambda_1^2 - \gamma)$$

We now set $\gamma = \beta\lambda_1^2$ to get a much simplified equation,

$$\Delta\mathbf{w}_1 = y_2\mathbf{x}_1\beta\lambda_1(1 - y_1^2)$$

which we use in the remainder of this paper.

The prior above is an improper prior ($\int P(\mathbf{w}_1|\mathbf{x}_1, H) = \infty$) but this prior may be made proper if we bound the region over which the weights have non-zero prior probability,

$$P(\mathbf{w}_1|\mathbf{x}_1, H) = \frac{1}{Z_w}\exp(\frac{1}{2}\gamma y_1^2) \tag{5}$$

for $|\mathbf{w}_{1i}| < a, \forall i$ and $P(\mathbf{w}_1|\mathbf{x}_1) = 0$ otherwise. Note that the value of a must be sufficiently large that the non-zero part of the posterior distribution lies within the prior's support. If a is fixed, the width of the posterior of y_2 is proportional to the width of the least probable region round the origin for the prior on y_1.

However we see that this method will not be successful in the linear case since the probability density function increases so strongly towards a that all weights will tend to a uniform vector all of whose elements are equal to a.

Finally, we note that using y_2 as the target density in this way is equivalent to the M step in the EM algorithm, while the adaption of the weights, $\mathbf{w}_2$ is equivalent to the E step. From the perspective of the y_2 neuron, these are reversed.

4.1 A Hierarchical Network

4.1.1 Random Dot Stereograms

Becker [2] has developed a data set which is an abstraction of random dot stereograms.

Each input vector consists of a one dimensional random strip which corresponds to the left image and a shifted version of this which corresponds to the

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

$\mathbf{w}_1$	0.07	**-6.49**	-0.003	**7.02**
$\mathbf{w}_2$	**-6.19**	-0.07	**5.34**	0.07
$\mathbf{w}_3$	0.14	**15.44**	0.25	0.11
$\mathbf{w}_4$	0.09	-0.13	**12.93**	0.21

Table 1: Examplar converged weights: $\mathbf{w}_1$ and $\mathbf{w}_2$ have identified the left shift and $\mathbf{w}_3$ and $\mathbf{w}_4$ have identified the right shift.

right image. The left image has components drawn with equal probability from the -1, 1 and the the right image is generated by choosing a randomly chosen global shift - either one pixel left or one pixel right - and applying it to the left image. Because the shifts are chosen with equal probability, there are two equal sets of correlations corresponding to left-shift and right-shift. In order to find these, we require two pairs of weight $(\mathbf{w}_1, \mathbf{w}_2)$ and $(\mathbf{w}_3, \mathbf{w}_4)$. The learning rules for $\mathbf{w}_3, \mathbf{w}_4$ are analagous to those for $\mathbf{w}_1$, $\mathbf{w}_2$; at each presentation of a sample of input data, a simple competition between the products $\mathbf{y}_1\mathbf{y}_2$ and $\mathbf{y}_3\mathbf{y}_4$ determine which weights will learn on the current input samples. Using a learning rate of 0.1 and 100 000 iterations, the weights converge to the vectors show in Table 1. The table clearly illustrates that these high correlations come from one pair learning the shift-left transformation while the other learns the shift-right. It should be noted at this point that Becker [1] only uses a subset of 12 of the 16 possible patterns. She removed those which are ambiguous (such as (-1,1,-1,1)) whereas we are drawing our data from all possible patterns. We are able to reliably find both correlations with a very simple network. However we require a second layer if we are to unambiguously identify the concepts of "left" and "right".

4.1.2 Factor Analysis

Factor Analysis is a statistical technique which attempts to explain/compress a data set into a small number of dimensions. Let us have an N-dimensional input vector at time, $\mathbf{y}$, and an M-dimensional output vector, $\mathbf{z}$, with $\mathbf{v_{ij}}$ being the weight linking input j to output i and let η be the learning rate. Then the operation of the network is given by

$$\mathbf{z}_i = \sum_{i=1}^{N} \mathbf{v}_{ij}\mathbf{y}_j, \quad \mathbf{z}_j(t+1) = \mathbf{y}_j(t) - \sum_{i=1}^{M} \mathbf{v}_{ij}\mathbf{z}_i, \quad \Delta v_{ij} = \eta \mathbf{y}_j(t+1)\mathbf{z}_i \quad (6)$$

It is well known that the above rules perform a Principal Component Analysis of the input data. It has recently been shown [3] that the addition of a nonlinearity at the outputs such as

$$\mathbf{a}_i = \sum \mathbf{v}_{ij}\mathbf{y}_j, \qquad z_i = 0.3log(1 + exp(\mathbf{a}_i - 3)/0.3) \quad (7)$$

gives a network which identifies the independent factors of the input data.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 445-452

$\mathbf{v}_1$	**0.91**	0	**0.91**	0	**0.91**	0
$\mathbf{v}_2$	**0.91**	0	**0.91**	0	**0.91**	0
$\mathbf{v}_3$	0	**0.91**	0	**0.91**	0	**0.91**
$\mathbf{v}_4$	0	**0.91**	0	**0.91**	0	**0.91**

Table 2: The converged weights of the Factor Analysis network. The weights $\mathbf{v}_1$ and $\mathbf{v}_2$ give output z_1, while $\mathbf{v}_3$ and $\mathbf{v}_4$ give output z_2, z_1 identifies left-shift while z_2 identifies right-shift.

We therefore train a CCA network with 12 pairs of output neurons each of which competes in pairs to get results similar to those of table 1. We then train a Factor Analysis Network on the output of the CCA network. In table2, we show the converged weights of a network in which we differentiate between the two sets of inputs from the CCA network.

$$a_i \quad = \quad \sum \mathbf{v}_{1ij}\mathbf{y}_{1j} + \sum \mathbf{v}_{2ij}\mathbf{y}_{2j}$$

The results showed conclusively that neuron $\mathbf{z_1}$ identified the left shift while neuron $\mathbf{z_2}$ identified the right shift.

5 Conclusion

We have previously [5] derived an artificial neural network as an implementation of the standard statistical technique of canonical correlation analysis and compared its capability with standard statistics methods on both real and artificial data.

In this paper, we have extended the network by implementing stronger constraints on the outputs and by using a constraint on the weight learning. we have also derived the same algorithm from a maximal likelihood criterion and shown that by using a specific prior on the network's weights, we can derive a simplified algorithm. We have shown the performance of this simplified algorithm on a standard problem which is an abstraction method is more biologically plausible than that use in [2] and is also far more effective and computationally simple.

References

[1] S. Becker. *An Information-theoretic Unsupervised Learning Algorithm for Neural Networks*. PhD thesis, Gradute Department of Computer Science, University of Toronto, 1992.

[2] S. Becker. Mutual information maximization: Models of cortical self-orgainization. *Network:Computation in Neural Systems*, 7:7–31, 1996.

[3] D. Charles and C. Fyfe. Modelling multiple cause structure using rectification constraints. *Network: Computation in Neural Systems*, 1998.

[4] P. L. Lai and C. Fyfe. Canonical correlation analysis using artificial neural networks. In *European Symposium on Artificial Neural Networks, ESANN98*, 1998.

[5] P. L. Lai and C. Fyfe. A neural network implementation of canonical correlation analysis. *Neural Networks*, 12(10):1391–1397, Dec. 1999.