EXPLORATORY CORRELATION ANALYSIS

Jos Koetsier, Donald MacDonald and Darryl Charles

Applied Computational Intelligence Research Unit,
University of Paisley,
Scotland
(koet-ci0, macd-ci0, char-ci0 @paisley.ac.uk)

ABSTRACT

We present a novel unsupervised artificial neural network for the extraction of common features in multiple data sources. This algorithm, which we name Exploratory Correlation Analysis (ECA), is a multi-stream extension of a neural implementation of Exploratory Projection Pursuit (EPP) and has a close relationship with Canonical Correlation Analysis (CCA). Whereas EPP identifies "interesting" statistical directions in a single stream of data, ECA develops a joint coding of the common underlying statistical features across a number of data streams. It has been shown that the principle of contextual guidance may be used to find a sparse coding of the features in dual natural image patches that is very different from single stream sparse coding experiments. The network only identifies those features which exist in both data streams and thus tend to be fewer in number and more complex in nature.

1. INTRODUCTION

In many real world situations, information is not available in a direct and clear way due to corruption of the signals. One approach to uncovering the inherent structure from these signals is to perform several measurements, possibly using different sensing techniques. By working on the principle that all of the signals share the same fundamental information, we may process the data in multi-streams in such a way that we identify significant features within streams that are also common between streams.

One method for extracting accurate information from multiple data sources is to use contextual guidance [1]. The underlying principle of this approach is that a neuron or processing unit not only uses information directly available to it, but also information about the context in which it operates. Neurophysical evidence appears to support the idea that the brain uses contextual guidance to obtain more accurate information about the surroundings.

In this paper we present a neural method capable of extracting features from different data sources and combining those to form a sparse joint coding. Other statistically based dual stream neural architectures have been proposed [2] [3] [4] but these tend to be based on second-order canonical correlation analysis. The method that we propose is capable of searching for higher order shared structure between data streams. Information theoretic based approaches have also been proposed which concentrate on the stereo disparity problem [5] or on contextual guidance [1].

We demonstrate the network with well-known artificial data for this area of research before applying it to two types of natural image data. The first of the natural image data experiments uses contextual data in that two neighbouring image patches are chosen while the second experiment uses stereo data. In the latter case there is considerable overlap between two image patches and there may also be nonlinear disparities. The resultant weight vectors for both of these experiments prove to be quite different.

2. EXPLORATORY PROJECTION PURSUIT

Before we outline the ECA algorithm, it is useful to explain the method on which it is based - Exploratory Projection Pursuit (EPP). EPP is a statistical technique that is used to visualise structure in high dimensional data. We project the data to a lower dimensional space which enables us to look for interesting structure manually. The projection should capture all of the aspects that we wish to visualise, which means it should maximise an index that defines a degree of 'interest' of the output distribution [6].

If the criterion of interest is determined by variance, we obtain the well-known principle components analysis. In this case, the criterion used to learn the basis vectors is determined entirely by the variance of the projected data. However, often the important features cannot be uncovered by taking only variance into account. Other measures must be defined that can help us to find the optimal basis set.

Another measure of interestingness is based on an argument that states that random projections tend to result in Gaussian distributions [7]. Therefore, we can define an interesting projection as one that maximises the non gaus-

sianity of the output distributions. Several measures of non gaussianity currently exist. In this paper we will concentrate on measures that are based on kurtosis and skewness.

2.1. Neural EPP

Our ECA network is most strongly related to the single stream, neural EPP algorithm based on the negative feedback framework [6]. The operation of the EPP network is outlined by (1) to (3). (1) describes the feed-forward step, in which the input values, $\mathbf{x}$, are multiplied by the weights, W , and summed to activate the output. This is followed by a feedback phase (2) in which the output values, $\mathbf{y}$, are fed back through the weights and subtracted from the input to form a residual, $\mathbf{r}$. This residual is then used in the weight update rule (3), where η is a learning parameter.

$$\begin{aligned} \mathbf{y} &= W\mathbf{x} && (1)\\ \mathbf{r} &= \mathbf{x} - W^T\mathbf{y} && (2)\\ \Delta W &= \eta \mathbf{r}^T \mathbf{f}(\mathbf{y}) && (3) \end{aligned}$$

The function $\mathbf{f}(\mathbf{y})$ in (3) causes the weight vectors to converge to directions that maximise a function whose derivative is $\mathbf{f}(\mathbf{y})$. Thus, if $\mathbf{f}(\mathbf{y})$ is linear, i.e. $\mathbf{f}(\mathbf{y}) = \mathbf{y}$, the EPP algorithm performs identically to Oja's subspace algorithm [8]. If the function is $\mathbf{f}(\mathbf{y}) \propto \mathbf{y}^2$, the third moment in the data is maximised and if $\mathbf{f}(\mathbf{y}) \propto \mathbf{y}^3$ is used, the fourth moment in the data is maximised.

For reasons of stability, the output functions are replaced by functions that have the same truncated Taylor Expansion. Instead of using $\mathbf{f}(\mathbf{y}) = \mathbf{y}^3$ the function $\mathbf{f}(\mathbf{y}) = -\tanh(\mathbf{y}) = \mathbf{y} - \frac{1}{3}\mathbf{y}^3 + \frac{2}{15}\mathbf{y}^5 - \ldots$ may be used.

3. EXPLORATORY CORRELATION ANALYSIS

We have extended the Neural EPP algorithm to allow for multiple input streams. Both streams are assumed to be have a set of common underlying factors. Mathematically we can write this as

$$\begin{aligned} \mathbf{y}_1 &= W\mathbf{x}_1 \\ \mathbf{y}_2 &= V\mathbf{x}_2 \end{aligned}$$

The input streams are denoted as $\mathbf{x}_1$ and $\mathbf{x}_2$, the projected data as $\mathbf{y}_1$ and $\mathbf{y}_2$ and the basis vectors are rows of the matrices W and V. Each input stream can be analysed separately by performing EPP and finding common statistical features that have maximum non-gaussianity. However, as we know that the features we are looking for have the same statistical structure, we can add another constraint which maximises the dependence between the outputs. This is depicted schematically in Figure 1.

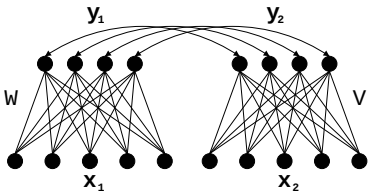


Fig. 1. Diagram of the ECA network

The simplest way to express this formally is by maximising $E(\mathbf{g}(\mathbf{y}_1)^T\mathbf{g}(\mathbf{y}_2))$. We also need to ensure the weights do not grow without bound, which we can achieve by adding weight constraints $W^TW = A$ and $V^TV = B$. Writing this as an energy function with Lagrange parameters $\lambda_{i,j}$ and $\mu_{i,j}$ [9] we obtain (4).

$$\begin{aligned} J(W,V) &= E(\mathbf{g}(W\mathbf{x}_1)^T\mathbf{g}(V\mathbf{x}_2)) + \\ &\quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_{i,j}(\mathbf{w}_i^T\mathbf{w}_j - a_{i,j}) + \\ &\quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\mu_{i,j}(\mathbf{v}_i^T\mathbf{v}_j - b_{i,j}) \qquad (4) \end{aligned}$$

The energy function (4) can be differentiated with respect to the weights $v_{i,j}$ and $w_{i,j}$. The maxima of (4) are:

$$\begin{aligned} \frac{\delta(J(W,V))}{\delta W} &= E((\mathbf{g}(\mathbf{y}_2)\otimes\mathbf{g}'(\mathbf{y}_1))\mathbf{x}_1^T) + \Lambda W = 0 \quad (5)\\ \frac{\delta(J(W,V))}{\delta V} &= E((\mathbf{g}(\mathbf{y}_1)\otimes\mathbf{g}'(\mathbf{y}_2))\mathbf{x}_2^T) + MV = 0 \quad (6)\\ WW^T &= A \quad (7)\\ VV^T &= B \quad (8) \end{aligned}$$

The $\otimes$ operator is defined as the element-wise multiplication of two vectors. The Lagrange multipliers can be calculated by multiplying (5) and (6) by W^T and V^T respectively. Inserting (7) and (8) results in:

$$\begin{aligned} \Lambda &= -E((\mathbf{g}(\mathbf{y}_2)\otimes\mathbf{g}'(\mathbf{y}_1))\mathbf{x}_1^T)W^TA^{-1}\\ M &= -E((\mathbf{g}(\mathbf{y}_1)\otimes\mathbf{g}'(\mathbf{y}_2))\mathbf{x}_2^T)V^TB^{-1} \end{aligned}$$

Reinserting these optimal Lagrange parameters into (5) and (6) yields:

$$\begin{aligned} \frac{\delta(J(W,V))}{\delta W} &= E((\mathbf{g}(\mathbf{y}_2)\otimes\mathbf{g}'(\mathbf{y}_1))\mathbf{x}_1^T)) - \\ &\quad E((\mathbf{g}(\mathbf{y}_2)\otimes\mathbf{g}'(\mathbf{y}_1))\mathbf{x}_1^T))W^TA^{-1}W\\ \frac{\delta(J(W,V))}{\delta V} &= E((\mathbf{g}(\mathbf{y}_1)\otimes\mathbf{g}'(\mathbf{y}_2))\mathbf{x}_2^T) - \\ &\quad E((\mathbf{g}(\mathbf{y}_1)\otimes\mathbf{g}'(\mathbf{y}_2))\mathbf{x}_2^T)V^TB^{-1}V \end{aligned}$$

Using stochastic gradient descent on these functions yields the following rules:

$$\Delta W = \eta[(\mathbf{g}(\mathbf{y}_2) \otimes \mathbf{g}'(\mathbf{y}_1))(\mathbf{x}_1^T - \mathbf{y}_1^T W)] \quad (9)$$
$$\Delta V = \eta[(\mathbf{g}(\mathbf{y}_1) \otimes \mathbf{g}'(\mathbf{y}_2))(\mathbf{x}_2^T - \mathbf{y}_2^T V)] \quad (10)$$

We have set the A and B matrices to the identity matrix, which causes the weights W and V to converge to orthonormal weight vectors.

As with the neural EPP algorithm, we need to replace the output functions with stable versions for the ECA algorithm. In contrast to the neural EPP algorithm, we not only require the derivative of the function to be maximised, but also the function itself. We therefore need an additional stable function, whose truncated Taylor expansion is $\mathbf{g}(\mathbf{y}) = \mathbf{y}^4$. The function we chose for the experiments in this paper is $\mathbf{g}(\mathbf{y}) = 1 - \exp(-\mathbf{y}^4)$.

3.1. Artificial data set

A simple experiment was performed to test the network. We used an artificial data-set, generated from a kurtotic and a normal data source. The inputs to the network are two three-dimensional input vectors as shown in Table 1. We used three types of data source, each with a different kurtosis value. Input S_1 and S_2 were generated by taking a value from a normal distribution and raising it to the power of 5. Input S_3 was generated from a normal distribution raised to the power of 3. The common data source S_3 is therefore less kurtotic than input S_1 or S_2. The last data source we used is S_4, which was taken from a normal distribution. In order to show the robustness of the network we added zero mean Gaussian noise with variance 0.2 to each of the inputs independently.

$$x_{1,1} = S_1 + N(0, 0.2) \quad x_{2,1} = S_2 + N(0, 0.2)$$
$$x_{1,2} = S_3 + N(0, 0.2) \quad x_{2,2} = S_3 + N(0, 0.2)$$
$$x_{1,3} = S_4 + N(0, 0.2) \quad x_{2,3} = S_4 + N(0, 0.2)$$

Table 1. Artificial data set. S_1 and S_2 are more kurtotic than the common source S_3. S_4 is a normal data source.

After training the network for 50000 iterations with a learning rate of 0.003, the weights converged to the values shown in Table 2 The network has clearly identified the common kurtotic data source and has ignored the common normal input and the independent input sources S_1 and S_2, although they are more kurtotic than S_3.

3.2. Dual Stream Blind Source Separation

In this section we describe an experiment, which is an adaption of the blind source separation problem. Instead of having one set of inputs, we generate two sets of inputs, which are both different linear mixtures of the same source signals. We used mixtures of three source signals, which were created artificially by randomly taking values from a normal distribution and raising them to the power of 3,

w	0.0029	1.0000	0.0028
v	0.0043	1.0000	-0.0182

Table 2. Weightvectors after training the ECA network on artificial data.

The mixing matrices, A and B, are shown below.

$$A = \begin{pmatrix} 2 & 5 & 1 \\ 5 & 2 & 9 \\ 9 & 2 & 3 \end{pmatrix} \qquad B = \begin{pmatrix} 3 & 6 & 1 \\ 9 & 4 & 7 \\ 1 & 5 & 3 \end{pmatrix}$$

To show the unmixing properties of the network, we examine $AC_{1,1}^{-1/2}W^T$ and $BC_{2,2}^{-1/2}W^T$.

$$AC_{1,1}^{-1/2}W_1^T = \begin{pmatrix} -0.0013 & -1.0006 & -0.0006 \\ 1.0002 & 0.0311 & -0.0038 \\ -0.0021 & -0.0036 & 1.0000 \end{pmatrix}$$

$$BC_{2,2}^{-1/2}W_2^T = \begin{pmatrix} -0.0021 & -1.0006 & -0.0017 \\ 1.0002 & 0.0313 & -0.0031 \\ -0.0021 & -0.0036 & 1.0000 \end{pmatrix}$$

These matrices show that combining the mixing, sphering and unmixing operations result in matrices that contain a one or minus one in each row. This indicates that the ECA algorithm has successfully unmixed the sources and has identified the common sources.

4. CONNECTION TO CCA

The linear one unit exploratory correlation analysis network is closely related to classical CCA. When the network is fully converged, the expected change in weights will be zero [10].

$$\begin{aligned} E(\delta\mathbf{w}) &= E(\eta y_2(\mathbf{x}_1^T - y_1\mathbf{w}^T)^T) \\ &= \eta E(\mathbf{v}^T\mathbf{x}_2\mathbf{x}_1^T - \mathbf{w}^T\mathbf{x}_1\mathbf{x}_2^T\mathbf{v}\mathbf{w}_1^T) = 0 \\ E(\delta\mathbf{v}) &= E(\eta y_1(\mathbf{x}_2^T - y_2\mathbf{v}^T)^T) \\ &= \eta E(\mathbf{w}^T\mathbf{x}_1\mathbf{x}_2^T - \mathbf{v}^T\mathbf{x}_2\mathbf{x}_1^T\mathbf{w}\mathbf{v}_2^T) = 0 \end{aligned}$$

Writing the term $\mathbf{w}^T\mathbf{x}_1\mathbf{x}_2^T\mathbf{v}$ as λ_1, $\mathbf{v}^T\mathbf{x}_2\mathbf{x}_1^T\mathbf{w}$ as λ_2, $E(\mathbf{x}_1\mathbf{x}_2^T)$ as $C_{1,2}$ and $E(\mathbf{x}_2\mathbf{x}_1^T)$ as $C_{2,1}$ we obtain:

$$\mathbf{v}^T C_{2,1} = \lambda_1 \mathbf{w}^T$$
$$\mathbf{w}^T C_{1,2} = \lambda_2 \mathbf{v}^T$$

and

$$C_{1,2}C_{2,1}\mathbf{w} = \lambda_1\lambda_2\mathbf{w}$$
$$C_{2,1}C_{1,2}\mathbf{v} = \lambda_1\lambda_2\mathbf{v}$$

When the network is stable, the weight vectors will therefore be eigenvectors of $C_{1,2}C_{2,1}$ and $C_{2,1}C_{1,2}$. Classical CCA however, requires the solutions to be eigenvectors of $C_{1,1}^{-1}C_{1,2}C_{2,2}^{-1}C_{2,1}$ and $C_{2,2}^{-1}C_{2,1}C_{1,1}^{-1}C_{1,2}$. The ECA network is capable of performing CCA, if the data-sources $\mathbf{x}_1$ and $\mathbf{x}_2$ are sphered prior to training the network by premultiplying them by $C_{1,1}^{-1/2}$ and $C_{2,2}^{-1/2}$, which causes $C_{1,1}$ and $C_{2,2}$, and therefore their inverses to become identity matrices. The resulting CCA weightvectors will be $C_{1,1}^{-1/2}\mathbf{w}$ and $C_{2,2}^{-1/2}\mathbf{v}$.

4.1. CCA Experiment

An experiment was carried out on a data set comprising 88 students' marks on five module exams [11]. The five modules can be divided in three open book exams and two closed book exams. The object of the experiment is to see how closely the ability to do an open book exam is correlated to the ability to do a closed book exam. We may also be interested in predicting the open book results from the closed book results.

We compare the results of three different CCA techniques. The first is the standard statistical CCA technique. This is assumed to be the most accurate and is used as a reference. The other two results are obtained by using the neural method as described by [2] and by using the ECA network. Both neural methods used 50,000 iterations. The CCA network used a learning rate 0.0001 and the ECA network used a learning rate of 0.0005. The results are shown in Table 3.

Standard Statistical CCA results			
$\mathbf{w}_1$	0.0260	0.0518	
$\mathbf{w}_2$	0.0824	0.0081	0.0035
Neural CCA			
$\mathbf{w}_1$	0.0264	0.0526	
$\mathbf{w}_2$	0.0829	0.0098	0.0041
ECA network			
$C_{1,1}^{-1/2}\mathbf{w}_1$	0.0258	0.0515	
$C_{2,2}^{-1/2}\mathbf{w}_2$	0.0826	0.0076	0.0032

Table 3. Results for the dataset.

4.2. Comparison to other CCA networks

Different implementations of CCA neural networks currently exist. Most of these networks have a more complicated structure as a result of a constraint that ensures the output variance is unity [2] [3]. The ECA network is derived with a weight constraint in mind rather than constraints on the outputs, which results in a fast, simple and robust network. The output constraint is achieved by sphering the data, which causes the covariance matrix of the input data to be the identity matrix. Because the weight vectors will converge to an orthonomal set, the output constraint will automatically be satisfied.

5. CONTEXTUAL GUIDANCE IN EARLY VISION

Natural images contain a great deal of structure, for example lines, surfaces edges and a variety of textures are present in most images. The human brain appears to have learned to code these structures efficiently. In an attempt to understand how our brains can achieve this we may adopt a statistical approach. From this perspective we can try to describe the relationship between neighbouring pixels. It has been argued that although there is a strong second order relationship between pixels, we cannot adequately capture the interesting structure that is necessary to analyse images efficiently by only considering second order statistics. Therefore a compact coding such as the well-known principal components analysis does not suffice.

Many arguments have been put forward that imply a more relevant code for natural images is a sparse coding [12], i.e. a coding which produces outputs with high kurtosis.

5.1. The EPP algorithm and sparse coding

As the EPP algorithm searches for codes with high kurtosis, it is suitable for coding natural images. Due to the large scale nature of the experiments, the performance may be improved by using a different form of weight constraint. In the case of EPP the weight update rule is simplified to a Hebbian update rule.

$$\Delta W = \eta \mathbf{x}^T \mathbf{f}(\mathbf{y})$$

and for ECA the weight update rules become:

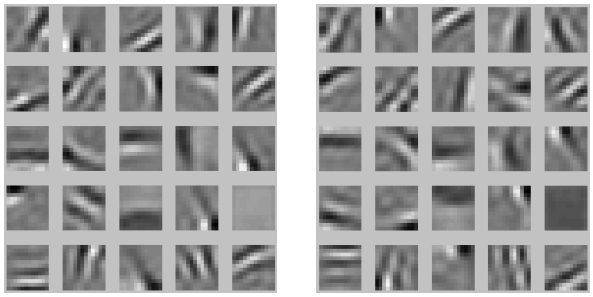
$$\begin{aligned} \Delta W &= \eta[(\mathbf{g}(\mathbf{y}_2) \otimes \mathbf{g}'(\mathbf{y}_1))\mathbf{x}_1^T] \\ \Delta V &= \eta[(\mathbf{g}(\mathbf{y}_1) \otimes \mathbf{g}'(\mathbf{y}_2))\mathbf{x}_2^T] \end{aligned}$$

As we have eliminated the weight constraints, the weights can grow indefinitely and two weights can learn the same feature. We ensure a bounded solution by using symmetric decorrelation given by:

$$W(t+1) = (W(t)W(t)^T)^{-\frac{1}{2}}W(t)$$

Because the EPP and ECA networks require data to be whitened we need to pre-process the images. First the data is mean-centered and then it is filtered by a filter with frequency response $R(f) = fexp(-(f/f_0)^4)$. This is a widely used whitening/low-pass filter that ensures the Fourier amplitude spectrum of the images is flattened. It also decreases the effect of noise by eliminating the highest frequencies.

Fig. 2. 10 converged weightvectors when training then EPP network with natural images



(a) Top view (b) Bottom view

Fig. 3. Converged weightvectors for the contextual guidance experiment on natural images

We carried out an experiment in which 9 natural images were chosen and preprocessed with the method described in section 5.1. We randomly sampled the pre-processed images by taking 12 by 12 pixel patches, which were used as inputs to the network. Figure 2 shows a sample of 10 weightvectors, after the network was fully trained. These results are similar to those obtained by other sparse coding networks [12] and have been related to the receptive fields of simple cells in the Striate cortex.

5.2. Contextual Guidance

The idea of contextual guidance in early visual processing can be simulated using the ECA network. For this experiment we chose 9 natural images, which we pre-processed as decribed in section 5.1. As before, we extracted 12 by 12 patches from random positions from a set of 9 natural images. The contextual information was added by taking the patch directly below the chosen patch and using both patches pair-wise as input to the ECA network. To facilitate convergence, both patches overlapped 2 pixels. The resulting weight vectors are displayed in Figure 5.2.

5.3. Discussion

The ECA network has formed a joint sparse code between two neighbouring patches. The weightvectors in Figure 5.2a and Figure 5.2b have the typical local wavelet like structure similar to those found in the one stream EPP experiment. Furthermore, the codes from both streams resemble each other in orientation and size, but differ in position. This indicates the network has formed both a sparse code, and a code that is related between two input patches.

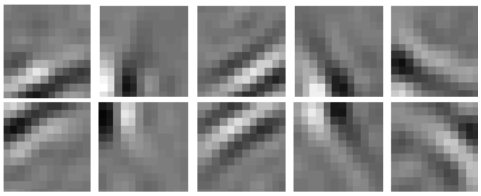


Fig. 4. Five pairs of matching converged weightvectors for the contextual guidance experiment on natural images

Another interesting observation is that most codes are formed on opposite sides of each patch. This is expected to occur at the sides where the two patches match and overlap slightly as the information shared between the inputs is highest there. Indeed most patches have formed this way but in the case of a few the opposite is true, which may be attributed to the orthogonality constaint.

To clarify the relationship between the patches, a sample of five pairs of converged weightvectors are show one over the other in Figure 4.

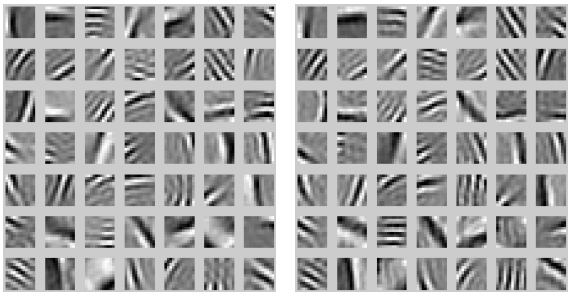
5.4. Stereo Images

An experiment related to the forming codes between neighbouring patches is the formation of common codes in stereo images. Stereo images consist of two images: one part as seen through the left eye and another as seen through the right. As both images are different views of the same scene, they share a number of features, which can be extracted with the ECA network.

For this experiment we chose 9 natural pre-processed stereo images. The images were sampled by randomly taking 12 by 12 patches, which were used pair-wise as input to the ECA network. The resulting weight vectors of the trained network are displayed in Figure 5.

5.5. Discussion of results

We have found a number of interesting differences between the filters obtained from standard images and those obtained from stereo images. The first difference is that there are significantly less shared codes for stereo images. This can be explained by the fact that stereo images are not only views of a scene at slightly different angles, but the two images are also shifted. This makes the 'overlap' between two patches smaller and the amount of shared information less. For two input streams of both 12 by 12 pixels, we extracted 49 components, which was experimentally determined to be the optimal number. Another difference is that the features found



(a) Left view (b) Right view

Fig. 5. Converged weightvectors when training the ECA network with stereo images

by the ECA network tend to have a wider variety of frequencies. Additionally, the features themselves tend to be larger.

When comparing the codes from both data streams, we can see many similarities in the codes, but there are also a number of interesting differences. A number of features are inverted versions of each other. This is a result of the positive only kurtotic objective function $\mathbf{g}(\mathbf{y}) = \mathbf{y}^4$. Also, a number of features are shifted, which can be attributed to the stereo disparity between the left and the right images.

Stereo images have been analysed before using sparse coding methods [13]. Usually, two patches from each image are taken and both patches are used simultaneously as input to the sparse coder. Our method differs from these, as ECA allows codes to form for each input stream independently, which are related through activations of the outputs.

6. CONCLUSION

We have presented a neural network based algorithm, ECA, which may be used to form a sparse coding of natural image samples across multiple data streams. The learned features represent a joint coding of the common underlying statistical features across the data streams. Because these features are shared between image streams, they are fewer in number and tend to be more complex in nature. In the future we intend to explore other algorithms within this framework and their application to image coding. The application of the network to areas of remote sensing may prove fruitful.

7. REFERENCES

[1] Jim Kay and W.A. Phillips, "Activation functions, computational goals and learning rules for local processors with contextual guidance," Tech. Rep. CCCN-15, Centre for Cognitive and Computational Neuroscience, University of Stirling, April 1994.

[2] P. L. Lai and C. Fyfe, "A neural network implementation of canonical correlation analysis," *Neural Networks*, vol. 12, no. 10, pp. 1391–1397, Dec. 1999.

[3] Z. Gou and C. Fyfe, "A family of networks which perform canonical correlation analysis," *International Journal of Knowledge-based Intelligent Engineering Systems*, April 2001.

[4] Tomas Landelius Magnus Borga, Hans Knutsson, "Learning canonical correlations," *SCIA*, 1997.

[5] Suzanna Becker, *An Information-theoretic Unsupervised Learning Algorithm for Neural Networks*, Ph.D. thesis, Gradute Department of Computer Science, University of Toronto, 1992.

[6] C. Fyfe, "A general exploratory projection pursuit network," *Neural Processing Letters*, vol. 2, no. 3, pp. 17–19, May 1995.

[7] Persi Diaconis and David Freedman, "Asymptotics of graphical projections," *The Annals of Statistics*, vol. 12, no. 3, pp. 793–815, 1984.

[8] E. Oja, "Neural networks, principal components and subspaces," *International Journal of Neural Systems*, vol. 1, pp. 61–68, 1989.

[9] Juha Karhunen and Jyrki Joutsensalo, "Representation and separation of signals using nonlinear pca type learning," *Neural Networks*, vol. 7, no. 1, pp. 113–127, 1994.

[10] John Hertz, Anders Krogh, and Richard G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley Publishing, 1992.

[11] K. V. Mardia, J.T. Kent, and J.M. Bibby, *Multivariate Analysis*, Academic Press, 1979.

[12] David J. Field, "What is the goal of sensory coding," *Neural Computation*, vol. 6, pp. 559–601, 1994.

[13] Patrick O. Hoyer and Aapo Hyvarinen, "Independent component analysis applied to feature extraction from color and stereo images," *Network: Computation in Neural Systems*, vol. 11, no. 3, pp. 191–210, 2000.