ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

Kernel Temporal Component Analysis (KTCA)

Dominique Martinez and Alistair Bray
LORIA-CNRS, BP 239,
54506 Vandoeuvre, France

Abstract.
We describe an efficient algorithm for simultaneously extracting multiple smoothly-varying non-linear invariances from time-series data. The method exploits the concept of maximizing temporal predictability introduced by Stone in the linear domain [11] - we term this temporal component analysis (TCA). Our current work extends this linear method into the non-linear domain using kernel-based methods [4]; it performs a non-linear projection of the input into an unknown high-dimensional feature space, computing a linear solution in this space. In this paper we describe the improved on-line version of this algorithm (KTCA) for working on very large data sets, and demonstrate its applicability for computer vision by extracting non-linear disparity directly from grey-level stereo pairs, without pre-processing.

1 Introduction

Independent component analysis (ICA) is a statistical method currently applied theoretically and practically to linear problems in many domains including pattern recognition. However in its basic form it is atemporal, and also limited in the non-linear domain: if any two variables X, Y are independent, $F(X), G(Y)$ are also independent. An alternative objective function which we consider here is to maximize temporal predictability in time series data [11, 12]. It is essentially temporal, and can be non-linear. We will refer to the linear method of Stone as TCA[11], and the non-linear extension of this we advance in this paper as KTCA. This is appropriate since they have precise analogues in PCA and KPCA [9], and have the same advantages of closed-form solutions avoiding local minima.
In previous work we extended TCA to the non-linear domain [4]. This used the kernel-based methods of Support Vector Machines that extended PCA to KPCA [9]. We develop this further here, providing an efficient online version of KTCA for very large data sets, that can extract multiple non-linear functions that are predictable in time. It extends Stone's linear algorithm into the non-linear domain [11], and his non-linear network algorithm into a closed-form method extracting multiple parameters [10]. It also advances on SFA [12], to the extent that it overcomes the curse of dimensionality through exploiting kernels. We present the results of running this algorithm on stereo pairs to extract non-linear disparity directly from grey levels, being the most predictable parameter.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

2 TCA and KTCA

First we summarize Stone's TCA [11]. Consider l time-series vectors $\mathbf{x}_{i<l}$ where each n-dimensional vector x_i is a linear mixture of n unknown temporally predictable components at time i. The problem is to find an n-dimensional weight vector $\mathbf{w}$ so that the output $y_i = \mathbf{w}^T\mathbf{x}_i$ at each i is a scaled version of a particular component. Many physical parameters exhibit such temporal predictability and overall variability: they are predictable over short horizons but unpredictable over long ones. Accordingly, a degree of predictability TP can be defined as the ratio between the long-term variance V and the short-term variance S of the output sequence i.e.

$$TP = \frac{V}{S} = \frac{\sum_i \overline{y_i}^2}{\sum_i \widetilde{y_i}^2} \tag{1}$$

where the values $\overline{y}_i$ and $\widetilde{y}_i$ represent the output at i centered using long- and short-term means. In TCA one aims to find the components that maximize TP, which can be rewritten as:

$$TP = \frac{\mathbf{w}^T\overline{\mathbf{C}}\mathbf{w}}{\mathbf{w}^T\widetilde{\mathbf{C}}\mathbf{w}} \; \textit{where} \; \overline{\mathbf{C}} = \frac{1}{l}\sum_i \overline{\mathbf{x}}_i\overline{\mathbf{x}}_i^T \; \textit{and} \; \widetilde{\mathbf{C}} = \frac{1}{l}\sum_i \widetilde{\mathbf{x}}_i\widetilde{\mathbf{x}}_i^T$$

Here, $\overline{\mathbf{C}}$ and $\widetilde{\mathbf{C}}$ are nxn covariance matrices estimated from the l inputs. The objective function TP is a version of the Rayleigh quotient and the problem to be solved is, in analogy to PCA, the right-handed generalized symmetric eigenproblem:

$$\overline{C}\mathbf{w} = \lambda\widetilde{C}\mathbf{w} \tag{2}$$

where λ is the largest eigenvalue and $\mathbf{w}$ the corresponding eigenvector. In this case, the component extracted $y = \mathbf{w}^T\mathbf{x}$ corresponds to the most predictable component with TP$= \lambda$. Most importantly, more than one component can be extracted by considering successive eigenvalues and eigenvectors which are orthogonal in the metrics $\overline{C}$ and $\widetilde{C}$, i.e. $\mathbf{w}_i^T\overline{\mathbf{C}}\mathbf{w}_j = 0$ and $\mathbf{w}_i^T\widetilde{\mathbf{C}}\mathbf{w}_j = 0$ for $i \neq j$.

KTCA: kernelized TCA

How can we make this algorithm non-linear? We first project the input data $\mathbf{x}$ into some unspecified high-dimensional feature space *via* a nonlinear mapping ϕ, and then find the weight vector $\mathbf{w}$ that maximizes TP in this space (see [4] for full derivations in this section). In this case, to optimize Eq. (2) the covariance matrices must be estimated in the feature space as

$$\overline{C} = \frac{1}{l}\sum_i \overline{\phi(\mathbf{x}_i)}\;\overline{\phi(\mathbf{x}_i)}^T \; \textit{and} \; \widetilde{C} = \frac{1}{l}\sum_i \widetilde{\phi(\mathbf{x}_i)}\;\widetilde{\phi(\mathbf{x}_i)}^T$$

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

where $\overline{\phi(\mathbf{x}_i)}$ and $\widetilde{\phi(\mathbf{x}_i)}$ represent the data centered in the feature space. The problem with this approach is that the dimensionality of the feature space can be huge [12]. Therefore, to avoid working with the mapped data directly, we assume that the solution $\mathbf{w}$ can be written as an expansion in terms of mapped training data: $\mathbf{w} = \sum_{i=1}^{l} \alpha_i \phi(\mathbf{x}_i)$. TP can then be written as:

$$TP = \frac{\alpha^T \overline{\mathbf{K}}\, \overline{\mathbf{K}}^T \alpha}{\alpha^T \widetilde{\mathbf{K}}\, \widetilde{\mathbf{K}}^T \alpha} \tag{3}$$

where $\alpha = (\alpha_1 \cdots \alpha_l)^T$ and $\overline{\mathbf{K}}$ (likewise $\tilde{\mathbf{K}}$) is a (lxl) matrix with entries defined as

$$\overline{\mathbf{K}}_{ij} = \phi(\mathbf{x}_i)^T\, \overline{\phi(\mathbf{x}_j)} \tag{4}$$

To avoid explicitly computing dot products in the feature space, we introduce kernel functions defined as $k(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$, which means we just have to evaluate kernels in the input space. Any kernel involved in Support Vector Machines can be used, e.g. linear, polynomial, RBF or sigmoid. By now defining the kernel matrix $\mathbf{K}$ with entries

$$\mathbf{K}_{ij} = k(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j) \tag{5}$$

we can arrive at the corresponding eigenproblem:

$$\overline{\mathbf{K}}\, \overline{\mathbf{K}}^T \alpha = \lambda\, \widetilde{\mathbf{K}}\, \widetilde{\mathbf{K}}^T \alpha \tag{6}$$

where λ is again the corresponding largest eigenvalue equal to TP. As for the linear case, more than one source can be extracted by considering successive eigenvalues and eigenvectors. Similarly to [5], a regularization on $||\alpha||^2$ that constrains the solution to have small α_i is obtained by replacing $\widetilde{\mathbf{K}}\widetilde{\mathbf{K}}^T$ by $\widetilde{\mathbf{K}}\widetilde{\mathbf{K}}^T + \mu\mathbf{I}$. In order to recover a temporal component, we need to compute the nonlinear projection $y = \mathbf{w}^T \phi(\mathbf{x})$ of a new input $\mathbf{x}$ onto $\mathbf{w}$ which is equivalent to $y = \sum_{i=1}^{l} \alpha_i k(\mathbf{x}_i, \mathbf{x})$.

3 Online KTCA

If the eigen problem is solved on the entire training set then the matrices (l x l) easily become computationally intractable. We propose a sparse solution using a shorter number p of training data in the expansion which we will call support vectors. The output is now $y = \sum_{i \in SV} \alpha_i k(\mathbf{x}_i, \mathbf{x})$ where SV is the set of support vectors. The solution must lie in a subspace spanned by SV, which is sensible if there is statistical redundancy in the dataset (guaranteed in our case by the temporal predictability assumption). The kernel elements $\mathbf{K}_{ij}$ are computed between the p support vectors $\mathbf{x}_i$ and the l training data $\mathbf{x}_j$. Thus, the kernel matrices $\mathbf{K}$, $\overline{\mathbf{K}}$ and $\widetilde{\mathbf{K}}$ are rectangular (p x l) but the covariance

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

matrices ($\overline{\mathbf{K}}\ \overline{\mathbf{K}}^T$) and ($\tilde{\mathbf{K}}\ \tilde{\mathbf{K}}^T$) used in the eigenproblem are only p x p. This approach can effectively solve very large problems, provided $p << l$.
However, the general question remains of how to choose support vectors. It is both necessary and sufficient that they span the space of the solution in feature space. If we use a linear kernel, we can coerce them to be the coordinates of the input space without considering the input patterns themselves. The algorithm is then identical to TCA (although any other basis set spanning the input space would be equivalent). However, in the non-linear case these vectors are chosen from the training set. They may be estimated by a kernel clustering algorithm in the feature space [1] or by a sparse greedy approximation algorithm [6]. Alternatively, as in Reduced Support Vector Machines [3], a random subset of the training data can be sufficient which showed that solutions built on a small set of support vectors chosen at random were usually better than those built on the entire dataset. We have found the same result, since a small number of support vectors provides an overconstrained, regularized solution. However, in the tests below, we choose supervectors such that for any two vectors x, y, $|k(x, y)| < \tau$. This ensures variation in the non-linear features ϕ. The number of support vectors is no longer a parameter: it is a function of τ. We found this method provides a significant improvement over random selection when large amounts of noise vectors (or "silence") are present in the data.
The algorithm requires minimal memory, making it ideal for very large data sets. It allows us to avoid a critical problem: computational limits mean we cannot have a sufficiently high ratio of data to model parameters, and the problem is underconstrained. We can always increase l to overcome this problem regardless of the input dimension (given enough different data), since memory is independent of l and computation time is linear in l. The greater the input dimension, the more complex is the non-linear feature space and hence the algorithmic power. This advantage results from two features. First, the implementation estimates the long- and short-term kernel means online using exponential time averages parameterized using half-lives Λ_s, Λ_l (see [10]). Second, the covariance matrices $\overline{\mathbf{K}}, \tilde{\mathbf{K}}$ are also updated online at each time step e.g. $\overline{\mathbf{K}}$ is updated by using the column vector of kernel values computed for the current time step; there is therefore no need to explicitly compute or store kernel matrices.

4 Simulations

We ran online KTCA on fragments of 128x128 stereo pair of the Pentagon shown in Figure 1[a](left). Running by pixel from left to right and top to bottom, we took 8 horizontal pixels from the left image and the corresponding 8 from the right to provide 16 inputs normalized to zero mean and unit variance. Repeating from top to bottom and left to right yields 30720 16-dimensional vectors. Setting $\tau = 0.3$ results in $SV = 132$ with an RBF kernel $\sigma = 1.5$; we set $\Lambda_s = 1, \Lambda_l = 100$. Each TP component can be reconstructed as an image: we show the first 4 of these in Figure 1[b]. The first component corresponds to non-linear disparity. To substantiate this affirmation, we simulated the stereo pair from the image used in [10] with the sub-pixel method described in [2], as shown in Figure 1[c](right). We define disparity as the 2D eggshell function in Figure 1[c](left) (max. disparity 1.4 pixels). Running KTCA on this

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

pair, the first TP component is as shown in Figure 1[d](middle): it correlates with the disparity $|r| > 0.98$. Most significantly, testing the solution obtained for the Pentagon on this artificial one, the first component correlates with disparity $|r| > 0.9$ as shown in Figure 1[c](right). Hence the parameter shown in Figure 1[b] is truly disparity.

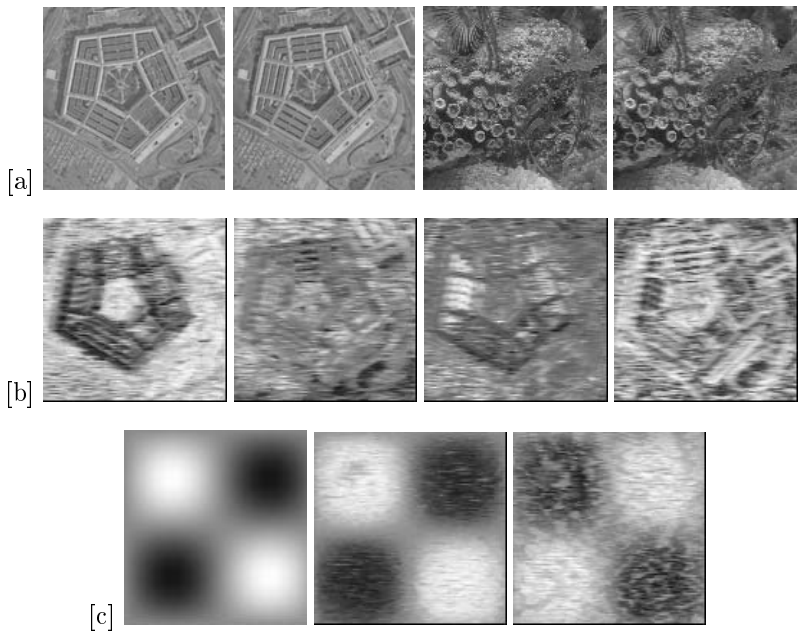

Figure 1: KTCA: the temporal components of stereo pairs. [a] Real stereo pair (left) and simulated one (right) [b] The four most predictable components of the Pentagon: the first corresponds to disparity. [c] The disparity function underlying the simulated pair (left); the most predictable component extracted on this pair (middle); the most predictable component extracted using the solution found for the Pentagon.

5 Conclusion

The method provides an efficient algorithm for maximizing a different statistical property to ICA which is both inherently spatio-temporal and potentially non-linear - temporal predictability. It overcomes the curse of dimensionality by projecting inputs into a non-explicit non-linear space defined by kernels, and finding the most predictable solution in this space. The result is an efficient closed-form solution. The online algorithm allows us to use large quantities of high-dimensional data, resulting in a complex feature space and an over-constrained solution. We test it on the hard task of extracting disparity from raw visual data confounded by other linear invariances, rather than simpler artificial random dot stereograms.

ESANN'2002 proceedings - European Symposium on Artificial Neural Networks
Bruges (Belgium), 24-26 April 2002, d-side publi., ISBN 2-930307-02-1, pp. 477-482

We suggest that whilst ICA seems most appropriate for creating sparse representations for tasks such as redundancy reduction found in early cortical areas, temporal statistics (and non-sparse representations) may turn out to be significant in higher cortical perceptual areas, such as visual IT, where cells have complex receptive fields providing invariance to transformations that are correlated over time [7, 8]; future work is proceeding in this direction.

References

[1] A. Ben-Hur, D. Horn, H. Siegelmann, and V. Vapnik. Support vector clustering. *Journal of Machine Learning Research*, to appear, 2001.

[2] S. Eglen, A. J. Bray, and J. V. Stone. Unsupervised discovery of invariances. *Network: Computation in Neural Systems*, 8(4):441–452, 1997.

[3] Y-J. Lee and O. L. Mangasarian. RSVM: Reduced support vector machines. In *Data Mining Institute Technical Report 00-07, July 2000. SIAM International Conference on Data Mining, Chicago, April 5-7, 2001, to appear.*

[4] D. Martinez and A. Bray. Nonlinear blind source separation using kernels. *IEEE Trans. Neural Networks*, Submitted, September 2000.

[5] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K-R. Muller. Fisher discriminant analysis with kernels. In *Proc. IEEE Neural Networks for Signal processing*, 1999.

[6] S. Mika, A.J. Smola, and B. Scholkopf. An improved training algorithm for kernel fisher discriminants. In *Proc. AISTATS, San Francisco. Morgan Kaufmann*, pages 98–104, 2001.

[7] D. I. Perret. Viewer-centred and object-centred coding of heads in the macaque temporal cortex. *Experimental Brain Research*, 86:159–173, 1991.

[8] M. Reisenhuber and T. Poggio. Models of object recognition. *Nature Neuroscience*, 3:1199–1204, November 2000.

[9] B. Schölkopf, A. Smola, and K-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[10] J. V. Stone. Learning perceptually salient visual parameters using spatiotemporal smoothness constraints. *Neural Computation*, 8(7):1463–1492, October 1996.

[11] J. V. Stone. Blind source separation using temporal predictability. *Neural Computation*, (13):1559–1574, 2001.

[12] L. Wiskott and T. Sejnowski. Slow feature analysis: Unsupervised learning of invariances. *Neural Computation*, 2001. (to appear).