LOCOCODE versus PCA and ICA

Sepp Hochreiter
Technische Universität München
80290 München, Germany

Jürgen Schmidhuber
IDSIA, Corso Elvezia 36
CH-6900-Lugano, Switzerland

Abstract

We compare the performance of three unsupervised learning algorithms on visual patterns that are mixtures of few underlying sources: "Independent Component Analysis" (ICA), "Principal Component Analysis" (PCA), and our new method "*Lo*w-*co*mplexity *co*ding and *de*coding" (Lococode). ICA and PCA fail to separate the sources no matter whether their number is known or not. Lococode, however, always separates them. It also codes with fewer bits per pixel than ICA and PCA.

1 Introduction

Recently several methods have been proposed for separating and extracting independent sources of given data: "Independent Component Analysis" (ICA, e.g. [3, 1, 2, 11]), methods enforcing sparse codes [4, 6, 12, 10], and "*lo*w-*co*mplexity *co*ding and *de*coding" (Lococode) [8, 9] based on *Flat Minimum Search* (FMS) [7]. Previous research already highlighted some of Lococode's advantages [8]. Here we experimentally compare ICA, "Principal Component Analysis" (PCA), and Lococode on visual data. Our criteria are: (1) Are the underlying statistical causes of the data discovered and separated? (2) What is the input reconstruction error? (3) How many bits per pixel are needed to code the input?

2 The compared methods

For PCA a standard MATLAB routine is used. ICA is realized by the JADE algorithm (Joint Approximate Diagonalization of Eigen-matrices, see [3]). JADE is based on whitening and subsequent joint diagonalization of 4th-order cumulant matrices. We used the MATLAB JADE version obtained via FTP from `sig.enst.fr`.

Lococode is realized by training a 3-layer autoassociator (AA) by *Flat Minimum Search* (FMS) [7]. Each layer is fully connected to the next. The hidden layer represents the code. FMS is a general, gradient-based regularization method for finding low-complexity networks (that can be described with few bits of information and require low weight precision) with low, tolerable training error. Such nets tend to exhibit high generalization capability. During learning FMS automatically prunes weights and units, and minimizes output sensitivity with respect to remaining weights and units. See [7] for details. It has been shown that FMS-based Lococode will result in *sparse* codes if inputs are describable by relatively few features (such as edges in images) [9].

3 Experiments

To measure the information conveyed by the various codes of the input data we train a standard backprop net on the training set used for code generation. Its inputs are the code components; its task is to reconstruct the original input. The average MSE on a test set is used to determine the reconstruction error.

Coding efficiency is measured by the average number of bits needed to code a test set input pixel. The code components are scaled to the interval $[0,1]$ partitioned into I discrete intervals — this results in I possible discrete values reflecting an input noise assumption (large $I \rightarrow$ little noise). Assuming independence of the code components we estimate the probability of each discrete code value by Monte Carlo sampling on the training set. To obtain the bits per pixels (Shannon's optimal value) on the test set we divide the sum of the negative logarithms of all code component probabilities (averaged over the test set) by the number of input components.

3.1 Experiment 1: noisy independent bars

We use a standard benchmark task: the input is a 5×5 pixel grid with horizontal and vertical bars at random, independent positions (10 possible bar locations). Each bar is activated with probability $\frac{1}{5}$. The inputs are noisy: pixels of activated bars randomly vary in $[0.1, 0.5]$. Input units not affected by currently active bars adopt activation -0.5. Then Gaussian zero mean noise with variance 0.05 is added to each input. The task is to extract the statistically independent features (the bars), and is adapted from [5, 6] but even more difficult because vertical and horizontal bars may be mixed in the same input.

Experimental conditions. The Lococode-trained AA has 25 input, 25 output, and 25 hidden units (HUs), although just 10 HUs are needed for optimal coding. Biased sigmoid output units are active in $[-1,1]$, HUs are active in $[0,1]$. Normal weights are initialized in $[-0.1, 0.1]$, bias weights with -1.0, the learning rate is 1.0. The net is trained on 500 randomly generated patterns for 5,000 epochs. $E_{tol} = 2.5$ (see [7]). The test set consists of 500 off-training set exemplars. For PCA and ICA, 1,000 training exemplars are used.

Lococode **results:** see Figure 1 and Table 1. 15 of the 25 HUs are pruned away. Lococode extracts an optimal (factorial) code which exactly mirrors the pattern generation process. It automatically finds the correct number of sources.

PCA and ICA results: see Figure 2 and Table 1. PCA codes and ICA-15 codes are unstructured and dense. For ICA-10 codes some sources are recognizable. They are not separated though: ICA and PCA fail to extract the true input causes and the optimal features. But at least PCA/ICA codes with 10

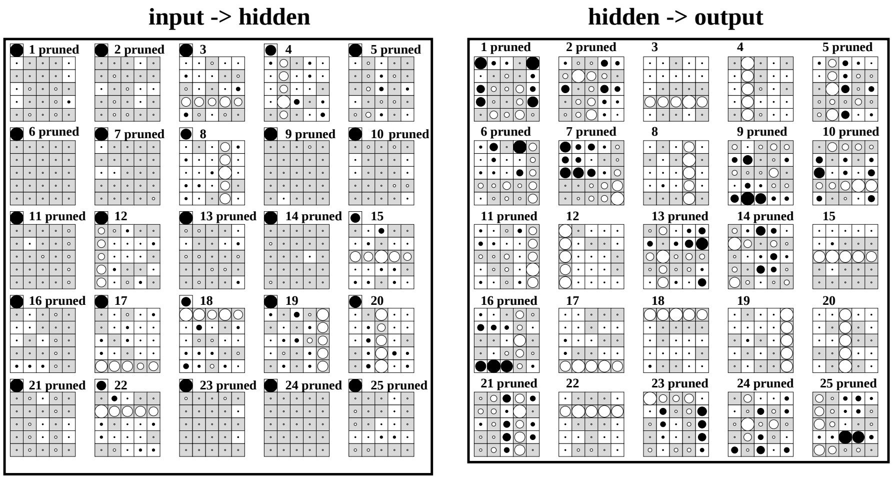

Figure 1: *Independent noisy bars. Left:* LOCOCODE*'s input-to-hidden weights. Right: hidden-to-output weights.*

components do convey as much information as 10-component codes found by LOCOCODE.

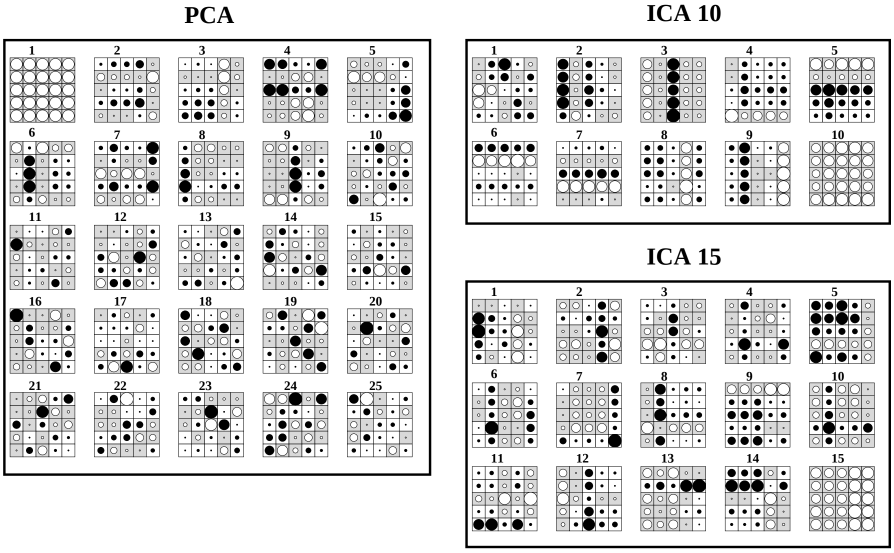

Figure 2: *Independent noisy bars. PCA and ICA: weights to code components (ICA with 10 and 15 components). Only ICA-10 codes reflect a few sources, but they do not achieve the quality of codes obtained through* LOCOCODE*.*

3.2 Experiment 2: village image

As in Experiment 1 the goal is to extract features from visual data, this time the aerial shot of a village. Figure 3 shows two images with 150 × 150 pixels, each taking on one of 256 gray levels. They are mostly dark except for certain white regions. 7 × 7 pixels subsections, corresponding to 49 inputs/outputs, from the left (right) image are randomly chosen as training (test) inputs, where gray levels are scaled to input activations in $[-0.5, 0.5]$. Targets are scaled to $[-0.7, 0.7]$.

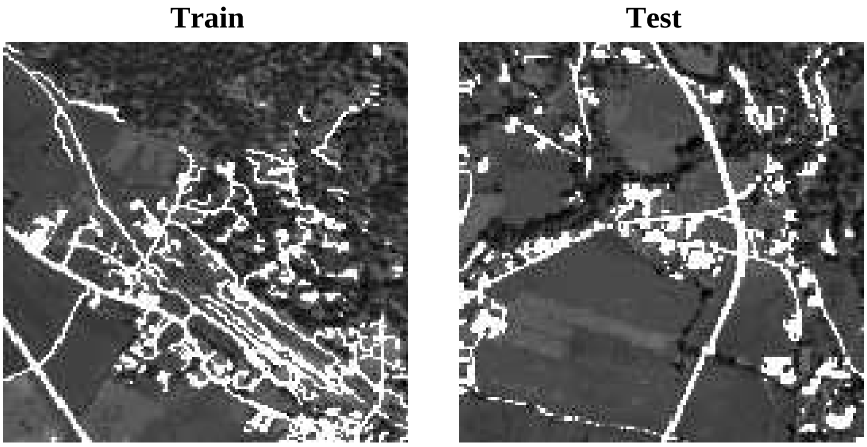

Figure 3: *Village image. Image sections used for training (left) and testing (right).*

Experimental conditions. Like in Experiment 1, except that training is stopped after 150,000 training examples, $E_{tol} = 3.0$. For PCA and ICA, 3,000 training exemplars are used.

Lococode **results:** see Figure 4 and Table 1. 9 to 11 HUs survive the 6 trials. The entire input is covered by white on-centers of surviving units that exhibit on-center-off-surround weight structures. This allows for detecting all white regions in the input field. Since most bright spots are connected, output/input units near an active output/input unit tend to be active, too.

PCA and ICA results: see Table 1. PCA-10 codes and ICA-10 codes are about as informative as 10-component codes found by Lococode. In fact, PCA's eigenvalues indicate that there are about 10 significant code components. Lococode automatically discovers this.

4 Conclusion

Lococode achieves success solely by reducing information-theoretic (de)coding costs. Unlike previous approaches it does not depend on explicit terms

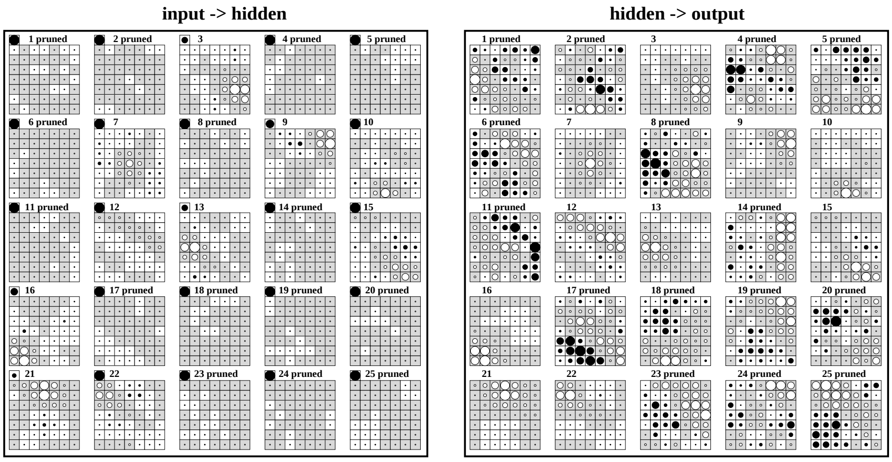

Figure 4: *Village. Left:* Lococode*'s input-to-hidden weights. Right: hidden-to-output weights. Most units are essentially pruned away.*

Exp.	input field	meth.	num. comp.	rec. error	code type	code efficency – reconst. 20	100
bars	5 × 5	LOC	10	1.05	sparse	0.84 - 1.15	1.37 - 1.06
bars	5 × 5	ICA	10	1.02	sparse	1.09 - 1.22	1.68 - 1.03
bars	5 × 5	PCA	10	1.03	dense	1.06 - 1.13	1.66 - 1.04
bars	5 × 5	ICA	15	0.71	dense	1.60 - 1.11	2.50 - 0.73
bars	5 × 5	PCA	15	0.72	dense	1.58 - 0.82	2.47 - 0.72
village	7 × 7	LOC	10	8.29	sparse	0.37 - 8.52	0.69 - 8.29
village	7 × 7	ICA	10	7.90	dense	0.46 - 8.44	0.80 - 7.91
village	7 × 7	PCA	10	9.21	dense	0.46 - 9.60	0.80 - 9.22
village	7 × 7	ICA	15	6.57	dense	0.70 - 7.40	1.20 - 6.58
village	7 × 7	PCA	15	8.03	dense	0.69 - 8.43	1.19 - 8.04

Table 1: *Overview over experiments: name of experiment, input field size, coding method, code size, reconstruction error, nature of code observed on the test set. PCA's and ICA's code sizes are prewired.* Lococode*'s, however, are found automatically. The final 2 columns show the coding efficiency measured in bits per pixels and the reconstruction error, for code components mapped to 20 and 100 discrete intervals.* Lococode *exhibits superior coding efficiency.*

enforcing independence or zero mutual information among code components, or sparseness.

Codes obtained by ICA, PCA and Lococode convey about the same information, as indicated by the reconstruction error. But Lococode's coding efficiency is much higher: it needs fewer bits per input pixel.

PCA does not separate data sources in the noisy bars experiment. ICA sometimes does, to a limited extent. LOCOCODE always does. Unlike ICA it does not need to know in advance the number of independent sources — it simply prunes superfluous code components: LOCOCODE seems more appropriate than ICA for visual coding tasks where few sources determine the input.

Acknowledgements. This work was supported by *DFG grant SCHM 942/3-1* from "Deutsche Forschungsgemeinschaft".

References

[1] S. Amari, A. Cichocki, and H.H. Yang. A new learning algorithm for blind signal separation. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 757–763. The MIT Press, Cambridge, MA, 1996.

[2] A. J. Bell and T. J. Sejnowski. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7(6):1129–1159, 1995.

[3] J.-F. Cardoso and A. Souloumiac. Blind beamforming for non Gaussian signals. *IEE Proceedings-F*, 140(6):362–370, 1993.

[4] P. Dayan and R. Zemel. Competition and multiple cause models. *Neural Computation*, 7:565–579, 1995.

[5] G. E. Hinton, P. Dayan, B. J. Frey, and R. M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268:1158–1161, 1995.

[6] G. E. Hinton and Z. Ghahramani. Generative models for discovering sparse distributed representations. Technical report, University of Toronto, Department of Computer Science, Toronto, Ontario, M5S 1A4, Canada, 1997. A modified version to appear in *Philosophical Transactions of the Royal Society* **B**.

[7] S. Hochreiter and J. Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[8] S. Hochreiter and J. Schmidhuber. Unsupervised coding with Lococode. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, editors, *Proceedings of the International Conference on Artificial Neural Networks, Lausanne, Switzerland*, pages 655–660. Springer, 1997.

[9] S. Hochreiter and J. Schmidhuber. Feature extraction through LOCOCODE. Technical Report FKI-222-97 (revised version), Fakultät für Informatik, Technische Universität München, 1998. Submitted to *Neural Computation.*

[10] M. S. Lewicki and B. A. Olshausen. Inferring sparse, overcomplete image codes using an efficient coding framework. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems 10*, 1998. To appear.

[11] L. Molgedey and H. G. Schuster. Separation of independent signals using time-delayed correlations. *Phys. Reviews Letters*, 72(23):3634–3637, 1994.

[12] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.