Binary Sparse Coding

Marc Henniges[1], Gervasio Puertas[1], Jörg Bornschein[1],
Julian Eggert[2], and Jörg Lücke[1]

[1] FIAS, Goethe-Universität Frankfurt am Main, Germany
[2] Honda Research Institute Europe, Offenbach am Main, Germany

Abstract. We study a sparse coding learning algorithm that allows for a simultaneous learning of the data sparseness and the basis functions. The algorithm is derived based on a generative model with binary latent variables instead of continuous-valued latents as used in classical sparse coding. We apply a novel approach to perform maximum likelihood parameter estimation that allows for an efficient estimation of all model parameters. The approach is a new form of variational EM that uses truncated sums instead of factored approximations to the intractable posterior distributions. In contrast to almost all previous versions of sparse coding, the resulting learning algorithm allows for an estimation of the optimal degree of sparseness along with an estimation of the optimal basis functions. We can thus monitor the time-course of the data sparseness during the learning of basis functions. In numerical experiments on artificial data we show that the algorithm reliably extracts the true underlying basis functions along with noise level and data sparseness. In applications to natural images we obtain Gabor-like basis functions along with a sparseness estimate. If large numbers of latent variables are used, the obtained basis functions take on properties of simple cell receptive fields that classical sparse coding or ICA-approaches do not reproduce.

1 Introduction

The mammalian brain encodes sensory stimuli by distributed activities across neural populations. Different neurons or different populations of neurons are hereby found to code for different aspects of a presented stimulus. Such distributed or factorial codes can (A) reliably encode large numbers of stimuli using relatively few computational elements and (B) they can potentially make use of the representation of individual components for further processing. In Machine Learning, factorial codes are closely associated with what is often called multiple-causes models. That is, they are related to probabilistic generative models which assume a data point to be generated by a combination of different hidden causes or *hidden variables*. Two very influencial models, that can be regarded as such multiple-causes models, are independent component analysis (ICA) [1] and sparse coding (SC) [2]. Indeed, since it was first suggested [2] sparse coding has become the standard model to explain the response properties of cortical simple cells. In its generative formulation, SC optimizes the parameters of a generative model with a sparse prior $p(\boldsymbol{s} \,|\, \lambda)$ and a Gaussian noise model $p(\boldsymbol{y} \,|\, \boldsymbol{s}, W, \sigma)$. In the last two decades, many different variants of the basic SC model have been introduced and discussed in the literature. These variants focused on different ways to train the parameters of the model (e.g., MAP estimates [2], sampling approaches [3] and many more). However, almost none of the approaches studied in the past estimated the data sparseness. This highlights that learning the sparseness seems a much more challenging task than learning the basis functions although usually just a single sparseness parameter has to be estimated. The learning algorithm studied in this paper will be shown to successfully estimate the sparseness. In applications to natural images we, furthermore, show that the algorithm reproduces simple cell properties that have only recently been observed [4].

2 Sparse Coding with Binary Hidden Variables

Consider a set of N independent data points $\{\boldsymbol{y}^{(n)}\}_{n=1,\dots,N}$ where $\boldsymbol{y}^{(n)} \in \mathbb{R}^D$ (D is the number of observed variables). For these data the studied learning algorithm seeks parameters $\Theta = (W, \sigma, \pi)$ that maximize the data likelihood $\mathcal{L} = \prod_{n=1}^{N} p(\boldsymbol{y}^{(n)} \,|\, \Theta)$ under the generative model:

$$p(\boldsymbol{s}|\pi) \;=\; \textstyle\prod_{h=1}^{H} \pi^{s_h} \big(1-\pi\big)^{1-s_h}\,, \qquad p(\boldsymbol{y} \,|\, \boldsymbol{s}, W, \sigma) \;=\; \mathcal{N}(\boldsymbol{y};\, W\boldsymbol{s}, \sigma^2 \mathbb{1})\,, \tag{1}$$

where $W \in \mathbb{R}^{D \times H}$ and H denotes the number of hidden variables s_h. For small values of π the latent variables are sparsely active. The basis functions $\boldsymbol{W}_h = (W_{1h}, \dots, W_{Dh})^T$ combine linearly and (given the latents) each observed variable y_d is independently and identically drawn from a Gaussian distribution with variance σ^2. The only difference to the generative model of classical sparse coding is thus the choice of binary latent variables (distributed according to a Bernoulli distribution) instead of latents with continuous values.

To optimize the parameters Θ, we use a variational EM approach (see, e.g., [5]). That is, instead of maximizing the likelihood directly we maximize the free-energy:

$$\mathcal{F}(q,\Theta) = \sum_{n=1}^{N} \Big[\sum_{\boldsymbol{s}} q^{(n)}(\boldsymbol{s}\,; \Theta^{\mathrm{old}}) \, \Big[\log\big(p(\boldsymbol{y}^{(n)} \,|\, \boldsymbol{s}, W, \sigma)\big) + \log\big(p(\boldsymbol{s} \,|\, \pi)\big) \Big] \Big] + H(q)\,, \tag{2}$$

where $q^{(n)}(\boldsymbol{s}\,; \Theta^{\mathrm{old}})$ is an approximation to the exact posterior and $H(q)$ denotes the Shannon entropy. In the variational EM scheme $\mathcal{F}(q,\Theta)$ is maximized alternately with respect to q in the E-step (while Θ is kept fixed) and with respect to Θ in the M-step (while q is kept fixed). Parameter update rules (M-step equations) are obtained by setting the derivatives of (2) w.r.t. the different parameters to zero. The obtained update rules contain expectation values such as $\langle \boldsymbol{s} \rangle_{q^{(n)}}$ and $\langle \boldsymbol{s}\boldsymbol{s}^T \rangle_{q^{(n)}}$ which are intractable for large H if $q^{(n)}$ is chosen to be the exact posterior ($q^{(n)}(\boldsymbol{s}\,;\Theta^{\mathrm{old}}) = p(\boldsymbol{s}\,|\,\boldsymbol{y}^{(n)},\Theta^{\mathrm{old}})$). To derive an efficient learning algorithm, our approach approximates the intractable expectation values by truncating the sums over the hidden space of $\boldsymbol{s}$:

$$\left\langle g(\boldsymbol{s})\right\rangle_{q^{(n)}} \;=\; \frac{\displaystyle\sum_{\boldsymbol{s}} p(\boldsymbol{s},\boldsymbol{y}^{(n)}\,|\,\Theta^{\mathrm{old}})\;g(\boldsymbol{s})}{\displaystyle\sum_{\tilde{\boldsymbol{s}}} p(\,\tilde{\boldsymbol{s}}\,,\boldsymbol{y}^{(n)}\,|\,\Theta^{\mathrm{old}})} \;\approx\; \frac{\displaystyle\sum_{\boldsymbol{s}\in\mathcal{K}_n} p(\boldsymbol{s},\boldsymbol{y}^{(n)}\,|\,\Theta^{\mathrm{old}})\;g(\boldsymbol{s})}{\displaystyle\sum_{\tilde{\boldsymbol{s}}\,\in\mathcal{K}_n} p(\,\tilde{\boldsymbol{s}}\,,\boldsymbol{y}^{(n)}\,|\,\Theta^{\mathrm{old}})}\,, \tag{3}$$

where $g(\boldsymbol{s})$ is a function of $\boldsymbol{s}$ (and potentially the parameters), and where $\mathcal{K}_n$ is a small subset of the hidden space. Eqn. 3 represents a good approximation if the set $\mathcal{K}_n$ contains most of the posterior probability mass. The approach will be referred to as *Expectation Truncation* and can be derived as a novel form of a variational EM approach (compare [6]). For other generative models similar truncation approaches have successfully been used [7, 8]. For the learning algorithm, $\mathcal{K}_n$ in (3) is chosen to contain hidden states $\boldsymbol{s}$ with at most γ active causes $\sum_h s_h \leq \gamma$. Furthermore, we only consider the combinatorics of $H' \geq \gamma$ hidden variables that are likely to have contributed to generating a given data point $\boldsymbol{y}^{(n)}$. More formally we define:

$$\mathcal{K}_n \;=\; \{\boldsymbol{s}\;|\;\big(\textstyle\sum_j s_j \leq \gamma \text{ and } \forall i \notin I : s_i = 0\big) \text{ or } \textstyle\sum_j s_j \leq 1\}, \tag{4}$$

where the index set I contains those latent indices h with the H' largest values of a *selection function* $\mathcal{S}_h(\boldsymbol{y}^{(n)})$. This function is given by:

$$\mathcal{S}_h(\boldsymbol{y}^{(n)}) = \tfrac{\boldsymbol{W}_h^{\mathrm{T}}}{||\boldsymbol{W}_h||}\,\boldsymbol{y}^{(n)}, \qquad \text{with } \;||\boldsymbol{W}_h|| = \sqrt{\textstyle\sum_{d=1}^{D}(W_{dh})^2}\,. \tag{5}$$

A large value of $\mathcal{S}_h(\boldsymbol{y}^{(n)})$ signals a high likelihood that $\boldsymbol{y}^{(n)}$ contains the basis function $\boldsymbol{W}_h$ as a component. The last term in (4) assures that all states $\boldsymbol{s}$ with just one non-zero entry are also evaluated. In numerical experiments on ground-truth data we can verify that for most data points the approach (3) with (4) and (5) indeed approximates the true expectation values with high accuracy. By applying this approximation, exact EM (which scales exponentially with H) is altered to an algorithm which scales polynomial with H' (approximately $\mathcal{O}\left(H'^{\gamma}\right)$) and linear with H. Note, however, that in general larger H also require larger amounts of data points.

With the tractable approximations for the expectation values $\left\langle g(\boldsymbol{s})\right\rangle_{q^{(n)}}$ computed with (3) to (5) the update equations for W and σ are given by:

$$W^{\mathrm{new}} = \left(\sum_{n\in\mathcal{M}} \boldsymbol{y}^{(n)}\,\langle\boldsymbol{s}\rangle_{q_n}^{T}\right)\,\left(\sum_{n'\in\mathcal{M}} \left\langle\boldsymbol{s}\,\boldsymbol{s}^{T}\right\rangle_{q_{n'}}\right)^{-1} \tag{6}$$

$$\sigma^{\mathrm{new}} = \sqrt{\frac{1}{|\mathcal{M}|\;D}\sum_{n\in\mathcal{M}}\left\langle\left|\left|\boldsymbol{y}^{(n)} - W\,\boldsymbol{s}\right|\right|^2\right\rangle_{q_n}} \tag{7}$$

Note that we do not sum over all data points $\boldsymbol{y}^{(n)}$ but only over those in a subset $\mathcal{M}$ (note that $|\mathcal{M}|$ is the number of elements in $\mathcal{M}$). The subset contains those data points for which (3) finally represents a good approximation. It is defined to contain the N^{cut} data points with highest values $\sum_{\tilde{\boldsymbol{s}}\in\mathcal{K}_n} p(\tilde{\boldsymbol{s}}, \boldsymbol{y}^{(n)} \,|\, \Theta^{\mathrm{old}})$, i.e., with the highest values for the denominator in (3). N^{cut} is hereby the expected number of data points that have been generated by states with less or equal γ non-zero entries: $N^{\mathrm{cut}} = N \sum_{\boldsymbol{s},\,|\boldsymbol{s}|\leq\gamma} p(\boldsymbol{s}\,|\,\pi) = N \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \pi^{\gamma'} (1-\pi)^{H-\gamma'}$.

Update equations (6) and (7) were obtained by setting the derivatives of Eqn. 2 (w.r.t. W and σ) to zero. Similarly, we can derive the update equation for π. However, as the approximation only considers states $\boldsymbol{s}$ with a maximum of γ non-zero entries, the update has to correct for an underestimation of π. If such a correction is taken into account, we obtain the update rule:

$$\pi^{\mathrm{new}} = \frac{A(\pi)\,\pi}{B(\pi)} \frac{1}{|\mathcal{M}|} \sum_{n\in\mathcal{M}} \langle |\boldsymbol{s}| \rangle_{q_n} \quad \text{with } |\boldsymbol{s}| = \sum_{h=1}^{H} s_h \text{ and} \tag{8}$$

$$A(\pi) = \sum_{\gamma'=0}^{\gamma} \binom{H}{\gamma'} \pi^{\gamma'} (1-\pi)^{H-\gamma'} \quad \text{and} \quad B(\pi) = \sum_{\gamma'=0}^{\gamma} \gamma' \binom{H}{\gamma'} \pi^{\gamma'} (1-\pi)^{H-\gamma'}.$$

Note that if we allow all possible states (i.e., $\gamma = H$), the correction factor $\frac{A(\pi)\,\pi}{B(\pi)}$ in (8) is equal to one over H and the set $\mathcal{M}$ becomes equal to the set of all data points (because $N^{\mathrm{cut}} = N$). Equation (8) then falls back to the exact EM update rule that can canonically be derived by setting the derivative of (2) w.r.t. π to zero (using the exact posterior). Also the update equations (6) and (7) fall back to their canonical form for $\gamma = H$. By choosing a γ between one and H we can thus choose the accuracy of the used approximation. The higher the value of γ the more accurate is the approximation but the larger are also the computational costs. For intermediate values of γ we can obtain very good approximations with small computational costs.

3 Numerical Experiments

The update equations (6), (7), and (8) together with approximation (3) define a learning algorithm that optimizes the full set of parameters of the generative model (1). In order to numerically evaluate the algorithm we ran several tests on artificial and natural data.

Linear bars test. We applied the algorithm to artificial bars data as shown in Fig. 1A. To generate this data we created $H = 10$ basis functions $\boldsymbol{W}_h$ in the form of horizontal and vertical bars. Each bar occupied 5 pixels on a $D = 5 \times 5$ grid. Bars were chosen to be either positive (i.e. $W_h^d \in \{0.0, 10.0\}$) or negative ($W_h^d \in \{0.0, -10.0\}$). Half of the basis functions was randomly assigned the negative values and the other half the positive values. Data points were generated by linearly superimposing these basis functions (compare, e.g., [9] for a similar task) with a sparseness value of $\pi H = 2.0$ (i.e., two active causes per image on average). To this data we added iid Gaussian noise (mean = 0.0, std = 2.0). After each trial we tested whether each basis function was uniquely represented

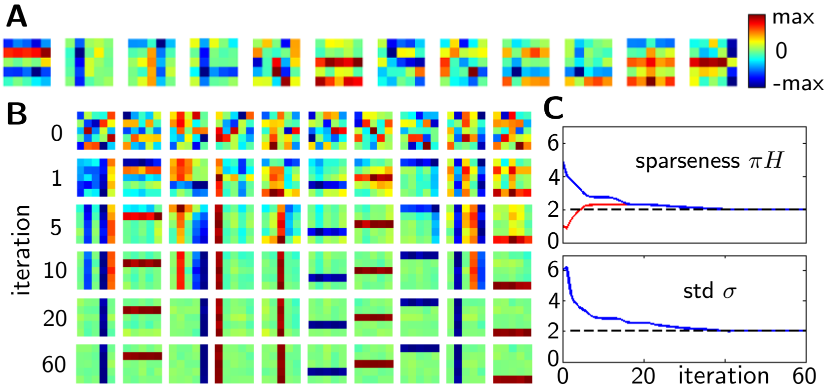

Fig. 1. Linear bars test with $H = 10$, $D = 5 \times 5$, and $N = 500$. **A** 12 examples for data points. **B** Basis functions for iterations given on the left. **C** Sparseness and standard deviation plotted over the iterations. Data for same experiment as in B in blue. Data for a run with initial sparseness value of 1.0 in red. Ground-truth indicated by dashed horizontal line.

by a single bar in oder to compute the success-rate, i.e. the reliability of the algorithm.

The approximation parameters were set to $\gamma = 3$ and $H' = 5$. We started with 20 iterations in which we set $|\mathcal{M}| = N$, then linearly decreased the amount of used data points in the next 20 iterations to $|\mathcal{M}| = N^{\text{cut}}$ where we kept it constant during the last 20 iterations, thus using a total of 60 iterations. The parameters W were initialized by drawing randomly from a Gaussian distribution with zero mean and a standard deviation of 2.0 (compare [6]). Sparseness was initialized at $\pi H = 5.0$, thus assuming that five of the causes contributed to an image on average. The standard deviation was initialized by calculating the sum over all squared data points which led to a value of $\sigma \approx 6.0$. After each iteration we added iid Gaussian parameter noise to the learned basis functions (mean $= 0.0$, std $= 0.05$).

We ran the algorithm with the above parameters 1000 times, each time using a newly generated set of $N = 1000$ data points. In 978 of these trials we recovered all bars ($\approx 98\%$ reliability) and obtained a mean value of $\pi H = 2.0$ (± 0.01 std) for the sparseness and $\sigma = 2.0 \pm 0.06$ for the data noise. Reliabilities increased when more data points were used (e.g., $\approx 99\%$ for $N = 4000$) and decreased for lower initial values of πH (e.g., $\approx 96\%$ and $\approx 84\%$ for $\pi H = 3$ and $\pi H = 1$, $N = 2000$, respectively). Figures 1B and 1C show the typical development of the parameters W, πH, and σ over the 60 iterations.

Natural image patches. In order to perform the experiment on natural images, we sampled $N = 200\,000$ patches of $D = 26 \times 26$ pixels from the van Hateren image database [10] (while constraining random selection to patches of images without man-made structures). As a form of preprocessing, we used

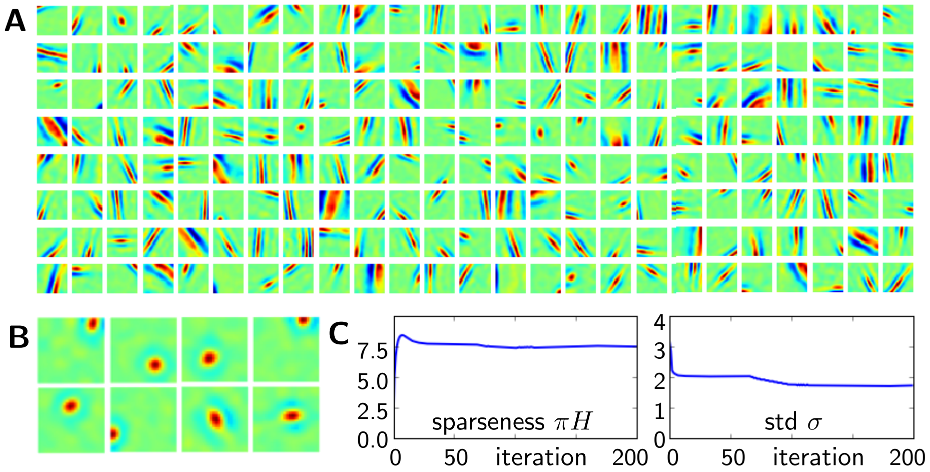

Fig. 2. Numerical experiment on image patches. **A** 200 basis functions randomly selected out of the $H = 700$ used. **B** The most globular of the $H = 700$ fields. **C** Time-courses of sparseness (πH) and data noise (in terms of standard deviation σ).

a Difference of Gaussians (DoG) technique[1]. According to the previous experiment, the initial condition for each basis function was set to the average over the preprocessed input patches plus small Gaussian white noise. The initial noise parameter σ was set following equation 7 by using all data points ($|\mathcal{M}| = N$). Finally, the initial sparseness value was taken to be $\pi H = 1$. The approximation parameters for the algorithm were set to $\gamma = 8$ and $H' = 10$. This choice reflects the relatively high average number of components that we expect for the relatively large patch size used (experiments with different γ and H values have all suggested an average of approximately 6 to 10 components per patch). The number of data points used was $|\mathcal{M}| = N$ during the first 66 iterations, decreased to $|\mathcal{M}| = N^{\mathrm{cut}}$ from iteration 66 to iteration 100 and kept at this value for the remaining 100 iterations. Fig. 2 shows the learned parameters for a run of the algorithm with $H = 700$ hidden variables. Fig. 2A shows a random selection of 200 of the 700 obtained basis functions. In Fig. 2B the most globular of the 700 basis functions are displayed. The monitored time-course of the data sparseness ($\pi\, H$) and the time-course of the data noise (σ) are displayed in Fig. 2C. As can be observed, we obtain Gabor-like basis functions with different orientations, frequencies, and phase as well as globular basis functions with no or very little orientation preferences (compare [4]). Along with the basis functions we obtain an estimate for the noise and, more importantly, for the data sparseness of $\pi\, H = 7.49$ active causes per 26×26 patch. Runs of the algorithm with H smaller than 700 (e.g. $H = 200$) resulted in similar basis functions. However, for smaller H, basis functions had the tendency to be spatially more constrained.

[1] Filter parameters were chosen as in [11]; before the brightest 2% of the pixels were clamped to the maximal value of the remaining 98% (influence of light-reflections were reduced in this way).

4 Discussion

We have studied a learning algorithm for sparse coding with binary hidden variables. In contrast to almost all the numerous SC and ICA variants, it is capable of learning the full set of parameters. To the knowledge of the authors there are only two previous SC versions that also estimate the sparseness: the algorithm in [3] which assumes a mixture of Gaussians prior, and the algorithm in [12] assuming a Student-t prior. To estimate the sparseness the approach in [3] uses sampling while the approach in [12] screens through discrete values of different sparseness levels to estimate the optimal one by comparison. In contrast, we use an update equation derived from a deterministic approximation (Expectation Truncation; [6]) which represents a novel form of variational EM. Another difference between the approaches [3] and [12] and our approach is the assumption of continuous-valued latents in those, and of binary latents in our case. Binary latents have frequently been used in the past ([13–15] and many more). The approach most similar to ours is hereby [15] which assumes the same underlying generative model. However, in none of these approaches the data sparseness is learned. The presented approach is thus the first algorithm that infers the appearance probabilities and data noise in a linear bars test (but compare [7] which learns the sparseness for non-linear superpositions with a different method). Also in applications to image patches, our approach estimates the sparseness in parallel to learning the basis functions and data noise (Fig. 2). The basis functions hereby take the form of Gabor-like wavelets and of globular functions with no or little orientation tuning. Interestingly, simple cell receptive fields that correspond to such globular functions were observed in *in vivo* recordings in [4]. Modelling approaches have only very recently reproduced such fields [16, 17, 11]. The system in [16, 11] is a neuro-dynamic approach that models cortical microcircuits. The model described in [17] is, notably, a SC version whose hidden variables have a more binary character than those of classical SC: they use latents with continuous values but explicitly exclude values in an interval around zero (while allowing zero itself). If applied to image patches, globular basis functions are obtained in [17] alongside Gabor-like basis functions. In that study the sparseness parameter had to be chosen by hand, however. The algorithm in [15] uses binary latents but, although applied to image patches, globular fields were not reported. This might be due to a relatively small number of hidden units used there. Also in [15] the sparseness level had to be set by hand.

Parameter optimization as studied in this paper can in future work be applied to SC models with continuous latents (e.g., with Laplacian or Student-t prior). Based on such models, the difference between binary and continuous latents can be investigated further. The observation that globular basis functions are obtained with the here presented algorithm might be taken as an indication that the assumption of binary or more binary latents at least facilitates the emergence of localized and circular symmetric basis functions. The observation that such globular functions also describe the response properties of many cortical simple cells [4] might have interesting implications for theories on neural coding.

Acknowledgement. We gratefully acknowledge funding by the German Federal Ministry of Education and Research (BMBF) in the project 01GQ0840 (BFNT Frankfurt), by the Honda Research Institute Europe GmbH, and by the German Research Foundation (DFG) in the project LU 1196/4-1. Furthermore, we gratefully acknowledge support by the Frankfurt Center for Scientific Computing.

References

1. Comon, P.: Independent Component Analysis, a new concept? Signal Process **36**(3) (1994) 287–314
2. Olshausen, B., Field, D.: Emergence of simple-cell receptive field properties by learning a sparse code for natural images. Nature **381** (1996) 607–609
3. Olshausen, B., Millman, K.: Learning sparse codes with a mixture-of-Gaussians prior. Proc NIPS **12** (2000) 841–847
4. Ringach, D.L.: Spatial Structure and Symmetry of Simple-Cell Receptive Fields in Macaque Primary Visual Cortex. J Neurophysiol **88** (2002) 455–463
5. Neal, R., Hinton, G.: A view of the EM algorithm that justifies incremental, sparse, and other variants. Learning in Graphical Models (1998) 355–369
6. Lücke, J., Eggert, J.: Expectation Truncation And the Benefits of Preselection in Training Generative Models. J Mach Learn Res, revision in review (2010)
7. Lücke, J., Sahani, M.: Maximal causes for non-linear component extraction. J Mach Learn Res **9** (2008) 1227–1267
8. Lücke, J., Turner, R., Sahani, M., Henniges, M.: Occlusive Components Analysis. Proc NIPS **22** (2009) 1069–1077
9. Hoyer, P.: Non-negative sparse coding. Neural Networks for Signal Processing XII: Proceedings of the IEEE Workshop (2002) 557–565
10. Hateren, J., Schaaf, A.: Independent Component Filters of Natural Images Compared with Simple Cells in Primary Visual Cortex. Proc Biol Sci **265**(1394) (1998) 359–366
11. Lücke, J.: Receptive Field Self-Organization in a Model of the Fine Structure in V1 Cortical Columns. Neural Computation (2009)
12. Berkes, P., Turner, R., Sahani, M.: On sparsity and overcompleteness in image models. Proc NIPS **20** (2008)
13. Hinton, G., Ghahramani, Z.: Generative models for discovering sparse distributed representations. Phil Trans R Soc B **352**(1358) (1997) 1177
14. Harpur, G., Prager, R.: Development of low entropy coding in a recurrent network. Network-Comp Neural **7** (1996) 277–284
15. Haft, M., Hofman, R., Tresp, V.: Generative binary codes. Pattern Anal Appl **6**(4) (2004) 269–284
16. Lücke, J., Sahani, M.: Generalized softmax networks for non-linear component extraction. In: ICANN 2007, LNCS. (2007) 657–667
17. Rehn, M., Sommer, F.T.: A network that uses few active neurones to code visual input predicts the diverse shapes of cortical receptive fields. J Comput Neurosci **22**(2) (2007) 135–146