

Orientation Selective Cells Emerge in a Sparsely Coding Boltzmann Machine

Cornelius Weber and Klaus Obermayer,
Informatik, Technische Universität Berlin, Email: cweber@cs.tu-berlin.de

Abstract

In our contribution we investigate a sparse coded Boltzmann machine as a model for the formation of orientation selective receptive fields in primary visual cortex. The model consists of two layers of neurons which are recurrently connected and which represent the lateral geniculate nucleus and primary visual cortex. Neurons have ternary activity values $+1$, -1 , and 0 , where the 0 -state is degenerate being assumed with higher prior probability. The probability for a (stochastic) activation vector on the net obeys the Boltzmann distribution and maximum-likelihood leads to the standard Boltzmann learning rule. We apply a mean-field version of this model to natural image processing and find that neurons develop localized and oriented receptive fields.

Introduction

Neurons in the primary visual cortical area V1 of mammals are strongly recurrently connected to a smaller number of LGN neurons. A typical V1 simple cell receives input from a small part of the visual field (localized receptive field) and often preferably responds to edges and lines of a certain orientation. Hubel and Wiesel [10] proposed that simple cells in cat striate cortex are orientation selective because they receive segregated ON- and OFF-input from appropriately elongated areas of the LGN retinotopic map. We model the development of this phenomenon. A possible other cause is recurrent intracortical interaction (e.g. [1]).

The activation statistics of V1 cells comes along with statistics of edges in natural images which is sparse [2]: neurons rarely en-

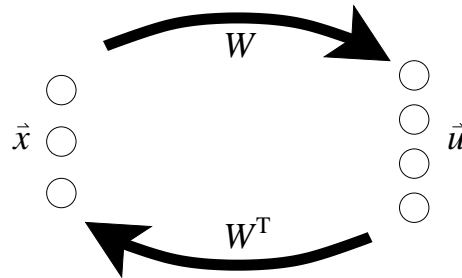


Figure 1: Architecture of the model. Input units with activations \vec{x} are fully connected to hidden units which have activations \vec{u} . Recurrent weights are depicted as separate feedforward, W , and feedback weights, W^T .

counter a matching edge. Therefore it has been suggested that sparseness constraints may be responsible for a stimulus driven generation of localized and oriented receptive fields.

Several models have been proposed to understand receptive field formation via the task to learn a sparse representation of visual images. A class of feedforward models applied for unsupervised data analysis, generate a sparse representation by maximizing the entropy on the hidden unit activity distribution [3][4]. It has been shown [5][11] that those models are equivalent to a recurrent type of data predicting model with sparse activation prior on hidden neurons (e.g. [7][12]). Accordingly, the role of the V1 \rightarrow LGN feedback is to predict (“extinguish”) a given data point by logically inhibitory feedback, i.e. antisymmetric weights. The role we propose of the feedback projection is to generate the data from hidden unit activations when there is no input.

Here we consider a recurrent stochastic neural network, a two layered Boltzmann machine, which learns to represent an en-

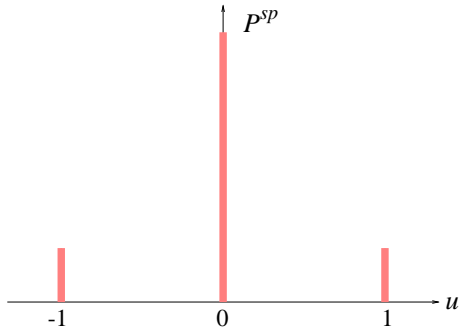


Figure 2: Prior density for the activation classes -1 , $+1$, and 0 of a hidden neuron derived from a flat prior over states. The latter value is n -times (here: $n = 5$) more probable than an active state.

semble of natural stimuli (Fig. 1). N input neurons are connected to H hidden neurons by symmetric forward and backward weights W , but no lateral connections are considered. Neurons are stochastic and can assume three different activity values $+1$, -1 (active states), and 0 (inactive state) (Fig. 2). Contrary to previous approaches, where sparseness is enforced via lateral competition [13] or where the mutual information between the “cortical” neurons is explicitly minimized [6], we impose sparseness by applying a high prior probability for the inactive state. In contrary to the competitive case the prior does not induce dependencies between hidden neuron activations. Given patches of natural images and a modified Boltzmann learning rule we find that a sparse representation, i.e. localized edge detectors emerge.

Theory and Methods

Probability distribution

In the Boltzmann machine framework, every activation state $\vec{s} = (\vec{u}, \vec{x})$ consisting of hidden unit activations \vec{u} and input neuron activations \vec{x} is characterized by an energy

$$E^{boltz}(\vec{s}) = -\frac{1}{2} \sum_j^N h_j x_j - \frac{1}{2} \sum_i^H h_i u_i,$$

where $h_j = \sum_i^H w_{ji} u_i$ denotes the net feedback input to an input neuron j and $h_i = \sum_j^N w_{ij} x_j$ denotes the net feedforward input to a hidden neuron i . The probability

that a given state occurs in the free-running phase is given by the Boltzmann distribution

$$P^-(\vec{s}) = \frac{e^{-\beta E^{boltz}(\vec{s})}}{\sum_{\{\vec{s}'\}} e^{-\beta E^{boltz}(\vec{s}')}} \quad (1)$$

In the clamped phase, where input neuron activations \vec{x} are given by a data point $\vec{\chi}^\mu$, with data index μ , we have:

$$P_{\vec{\chi}^\mu}^+(\vec{u}) = \frac{e^{-\beta E^{boltz}(\vec{\chi}^\mu, \vec{u})}}{\sum_{\{\vec{u}'\}} e^{-\beta E^{boltz}(\vec{\chi}^\mu, \vec{u}')}} \quad (2)$$

Note that the probability is defined over all states \vec{s} and takes into account the degeneracy of the inactive state.

Cost function and learning rule

Let $P(\vec{\chi}^\mu)^-$ denote the probability that the Boltzmann machine generates an observed data point $\vec{\chi}^\mu$ in the free-running phase. For a set of (statistically independent) data points $\{\vec{\chi}^\mu\}$ we obtain the log-likelihood

$$\mathcal{L}(\{\vec{\chi}^\mu\}) = \sum_{\mu} \ln P(\vec{\chi}^\mu) \quad (3)$$

of their generation which - in the limit of an infinite number of observations - becomes

$$\mathcal{L}(\{\vec{\chi}^\mu\}) = \sum_{\vec{\chi}^i} P^+(\vec{\chi}^i) \ln P^-(\vec{\chi}^i) \quad (4)$$

where the sum extends over all possible input activations. Maximizing Eq. (4) by gradient ascent we obtain [8]:

$$\Delta w_{ij} = \varepsilon \left(\sum_{\{\vec{\chi}\}} \sum_{\{\vec{u}\}} P_{\vec{\chi}}^+(\vec{u}) u_i \chi_j - \sum_{\{\vec{x}\}} \sum_{\{\vec{u}\}} P^-(\vec{x}, \vec{u}) u_i x_j \right) \quad (5)$$

for the architecture depicted in Fig. 1. ε denotes the learning step size. The two Hebbian/anti-Hebbian terms for learning w_{ij} are the expectation values of the correlation between activations u_i and χ_j (clamped phase) or u_i and x_j (free running phase).

Sparseness

In order to impose sparseness we consider n “zero”-states with degenerate energy besides the binary states $+1$ and -1 of the standard Boltzmann machine. We now

have multiple states with identical activations. The Boltzmann distribution Eq. (2) remains unchanged if all states are distinguished (by some label different from activations). The sum in the denominator (partition function) counts all states once (by the label which distinguishes them all).

A different point of view is to take into account multiplicity of identical states by a *prior* probability $P^{sp}(\vec{s})$ for a sparse activation. The Boltzmann distribution becomes

$$P_{\vec{s}} = \frac{P^{sp}(\vec{s}) e^{-\beta E^{boltz}(\vec{s})}}{\sum_{\{\vec{s}'\}} P^{sp}(\vec{s}') e^{-\beta E^{boltz}(\vec{s}')}}$$

where in the partition function the sum extends over states \vec{s}' with different activation values only. Fig. 2 shows this prior distribution for the three activation values of one neuron. Its Fisher-kurtosis is

$$\frac{\sum_{\{u\}} P^{sp}(u) u^4}{(\sum_{\{u\}} P^{sp}(u) u^2)^2} - 3 = \frac{n}{2} - 2.$$

The kurtosis is a measure of sparseness as it increases with the number of inactive neurons. It becomes positive for $n > 4$. It has been argued that distributions with positive kurtosis are suited for independent component analysis on sparsely distributed data [3] and produce sparse code [12].

Stochastic neurons

The probability for a hidden neuron i to have activity ± 1 or 0 is

$$\begin{aligned} P(u_i = \pm 1) &= \frac{1}{Z_i} e^{\mp \beta h_i} \\ P(u_i = 0) &= \frac{n}{Z_i} \end{aligned} \quad (6)$$

where the normalizing constant considers its possible values -1 , n -times zero and $+1$:

$$Z_i = e^{\beta h_i} + n + e^{-\beta h_i}$$

Figure 3 shows these corresponding transfer functions of stochastic neurons.

Mean activations

Given an input h_i to a hidden neuron i , its mean activation $\langle u_i \rangle = \sum_{\{\vec{u}\}} P(\vec{u}) u_i$ is the weighted average over activations $+1$, 0 and -1 and leads to an effective transfer function of a continuous neuron (Fig. 4):

$$\langle u_i \rangle = \varphi_u(h_i) = \frac{e^{\beta h_i} - e^{-\beta h_i}}{e^{\beta h_i} + n + e^{-\beta h_i}} \quad (7)$$

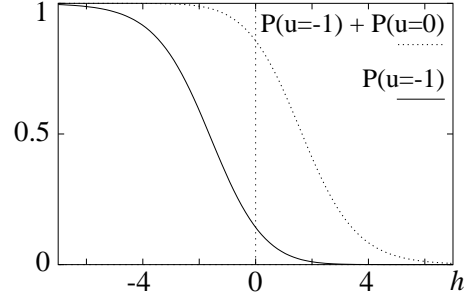


Figure 3: $P(u_i = -1)$, left curve, and $1 - P(u_i = +1)$, right curve, as a function of the net-input h , here for $n = 5$. The vertical distance between both curves is the probability for 0 activation.

Input neuron activations need to cover a wide continuous range for the purpose to generate our data. We assume a (infinite) homogeneous prior on equidistantly spaced discrete input activations. Taking the mean activation $\langle x_j \rangle = \sum_{\{\vec{x}\}} P(\vec{x}) x_j$, given an input h_j to an input neuron j , we obtain the identity transfer function

$$\langle x_j \rangle = \varphi_x(h_j) = h_j \quad (8)$$

Neuron update dynamics

Considering on-line learning, the left term of Eq. (5) can be written as

$$\varepsilon \sum_{\mu} \chi_j^{\mu} \langle u_i \rangle^+ \quad (9)$$

where $\bar{\chi}^{\mu}$ denote clamped stimuli and $\langle u_i \rangle^+ = \sum_{\{\vec{u}\}} P_{\bar{\chi}^{\mu}}^+(\vec{u}) u_i$. In the clamped phase, feedback has no effect. We find the exact mean activation of a hidden neuron by one feedforward computation.

For the free-running phase, the network architecture without lateral weights encourages an alternating update of the input and the hidden layer. Following the mean-field approximation [9], we write the right term of Eq. (5) as

$$-\varepsilon \sum_{\{\vec{u}\}} P_{\vec{x}}^-(\vec{u}) u_i \sum_{\{\vec{x}\}} P_{\vec{u}}^-(\vec{x}) x_j. \quad (10)$$

Eq. (10) is approximated by alternately updating all hidden neurons (Eqs. (6) or (7)) and all input neurons (Eq. (8)).

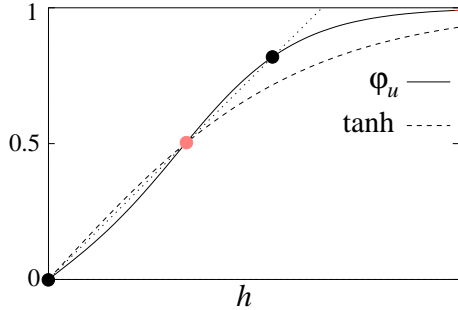


Figure 4: Transfer functions $\tanh(h)$ (dashed line) and the sparse mean-field transfer function $\varphi_u(h)$ (solid, curved line), plotted for $n = 10$. Only the positive half-axis is shown and both functions are scaled such that they intersect the identity function (dotted line) at 0.5. A single-neuron dynamics $h(t+1) = \tanh(h(t))$ exhibits the stationary states 0.5 (stable) and 0.0 (unstable). The dynamics $h(t+1) = \varphi_u(h(t))$ on the other hand exhibits two stable fixed points at 0.0 and $\varphi_u(h) = 0.8$, and an unstable fixpoint at 0.5.

Weight constraint

As an effect of sparse hidden neuron activations the data are under-estimated: In the free-running phase hidden neurons generate activity values on the input units which are smaller than values given by the data. The weights compensate via increasing their absolute values and - as a result - the net input h_i of any hidden neuron i will also become larger. As a consequence more neurons will take part in data generation and they will decide more decisively for the saturating states $+1$ or -1 of their transfer function and sparse coding is lost.

In order to prevent weights from becoming too large we consider a Bayesian weight prior and maximize the posterior probability $P(W|\{\bar{\chi}^\mu\}) \approx \mathcal{L}(\{\bar{\chi}^\mu\})P(W)$ where

$$P(\vec{w}_i) = e^{-\frac{d_w}{4} \sum_{j'}^N w_{ij'}^2 \sum_{j''}^N w_{ij''}^2}.$$

d_w is a scaling factor. Gradient ascent leads to the additive correction term to the above Boltzmann learning rule Eq. (5), i.e.

$$\Delta w_{ij}^{const} \approx -d_w w_{ij} \sum_{j'}^N w_{ij'}^2. \quad (11)$$

This soft constraint on a hidden neuron weight vector is local in the sense that it does not depend on any weight of any other hidden neuron. Thus, hidden neurons do not interact via this constraint. Weight vectors among input neurons do, but the Bayesian prior on the weights is separate from the Boltzmann machine framework and in particular does not influence activation dynamics.

Results

We took 16 greyscale images of natural scenes where for each image pixel values were normalized by subtraction of their mean and division by their variance. A data point $\bar{\chi}$ was generated by selecting an image randomly and cutting out a random part of the size of the input layer. The mean input value was subtracted of these pixels.

Weights were initialized with small random values with mean zero. Then the following on-line learning procedure was repeated 1000000 times. One clamped relaxation step was done with a data point to compute Eqs. (7, 9). One free-running relaxation consisted of 16 iterations where Eqs. (8, 10) were paired with either Eqs. (6) for stochastic hidden neurons or with Eq. (7) for continuous mean-field units. In the case of stochastic hidden neurons the mean used for Hebbian learning was taken from the last 8 iterations, in the case of mean-field hidden neurons the values of the last step were taken as the best approximation to the real mean. Then, a weight update was made (Eqs. (5) and (11)).

The parameters were: learning stepsize $\varepsilon = 0.001$ lowered to 0.0001, number (degeneracy) of zero-activity states of a hidden neuron $n = 15$, constraint on the weights $d_w = 0.1$, inverse temperature $\beta = 1.5$.

Edge detectors

Fig. 5 shows the weights after training. The characteristics of the receptive fields changes with the length of the weight vectors. Neurons with large weights receive strong input and are more often active. Lacking sparseness they extract features resembling principal components. The majority of smaller vectors form higher frequency edge detectors which are well localized within the input field. If the input images consist of

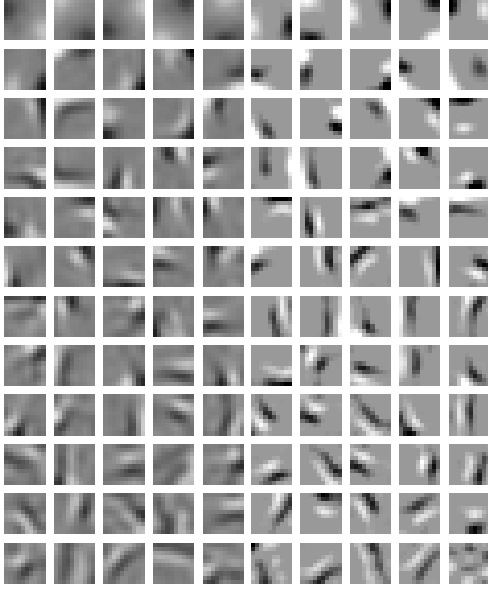


Figure 5: The weight matrix after training. Each square shows the receptive field of one of the 12×10 hidden neurons, black indicating negative, white positive weights to one of the 10×10 input neurons. On the left half, brightness scales linearly with the weights. On the right half, contrast is sharpened by a piecewise linear function. Fields are ordered left-to-right, top-to-bottom in the order of the length of the weight vector. A larger number of hidden neurons than input neurons allow for an overcomplete representation of the input.

white noise only, then there is no structure to be extracted. Receptive fields remain unstructured.

Clamped and free-running phase

Fig. 6, left, shows a receptive field after only the clamped phase was used for learning. All hidden neurons see the same data and do not interact. Identical fields develop (except for the sign) and correspond to the first principal component of the data as expected (cf. [8]).

The right receptive field of Fig. 6 shows a training result if only the free running phase was used. To prevent all weights to become zero we changed the weight constraint Eq. (11) to $\Delta w_{ij}^{const'} \approx +d_w w_{ij} / (\sum_{j'} w_{ij'}^2)$. Using $d_w = 0.1$, weights were comparable in size to those of the normally trained net.

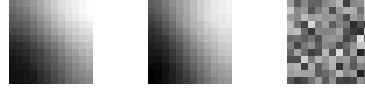


Figure 6: Left: a receptive field of a net which had been trained during clamped phase only. Middle: the receptive field of an alonstanding linear neuron. Right: one receptive field of a net which had been trained during the free-running phase only.

Field-vectors of hidden cells were maximally different, i.e. perpendicular, if there was no overcomplete coding. Comparing this result with the clamped case we see that the only interaction between hidden neurons occurs through the feedback loop in the free running phase.

Generation of images

Fig. 7 shows snapshots of activations on the input neurons of four differently trained networks as they relaxate in a prolonged free running phase.

After 16 iterations in the free running phase as performed for training, the activations of mean-field hidden neurons (Eq. (7)) have still not converged. Instead, the upper row of Fig. 7 shows that input values converge to stronger values after time. Then, rarely a hidden neuron remains “off”. Obviously, the fixpoint at zero (see Fig. 4) is not stable in the case of multiple neurons.

The second row of Fig. 7 shows input unit activations as they occur in the case of stochastic hidden neurons (Eqs. (6)). They fluctuate, thus hidden neurons do not get stuck in an all-active state.

The network which has been trained during the clamped phase generates images which resemble the first principal component of the data (third row). The activation patterns of a network which was trained during the free-running phase only converge to zero with a weak change of their appearance (forth row). Considering only the direction of the activation vector, we thus find a variety (near continuity) of attractors.

Discussion

For the purpose of biological modeling, we can eliminate negative activation values of

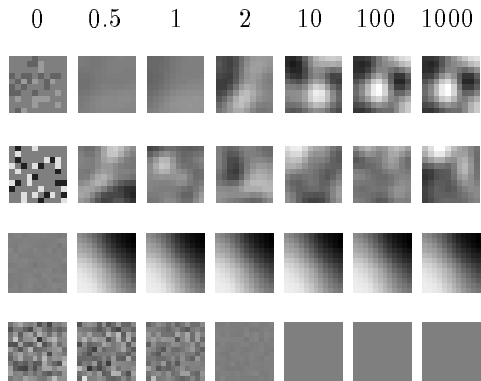


Figure 7: Each row shows the activation pattern across input units during the free running phase at seven different time-steps, given on top in multiples of the relaxation time used for learning. Except for the second row, mean-field hidden neurons (Eq. (7)) trained on 8 free relaxations were used. First row: network trained by the full Boltzmann learning rule; second row: like first row but with stochastic hidden neurons (Eqs. (6)) trained on 16 relaxations; third row: network trained during the clamped phase only; last row: network trained during the free running phase only.

hidden neurons if Eqs. (6) are changed to

$$P'(u_i = +1) = \frac{e^{\beta h_i}}{Z'_i}, \quad P'(u_i = 0) = \frac{n}{Z'_i}$$

with normalizing constant $Z'_i = e^{\beta h_i} + n$. Simulation results are comparable to the results shown without change of parameters.

Negative activation values on the input neurons are identified with positive activations of OFF-layer LGN cells. The sign cancels out by feeding these through the negative weights which are considered as (positive) connections to the OFF-layer.

The role of the feedback weights in our model to generate input data contradicts with their role to “extinguish” a given data point in other models (eg. [12]). The biological strategy remains to be discovered.

References

[1] P. Adorjan, J.B. Levitt, J.S. Lund, and K. Obermayer. A model for the intracortical origin of orientation preference and tuning in macaque striate cortex. *Visual Neuroscience*, 16:303–318, 1999.

[2] H. Barlow. *Large Scale Neuronal Theories of the Brain.*, chapter What is the computational Goal of the Neocortex?, pages 1–22. MIT Press, 1994.

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

[4] Anthony J. Bell and Terrence J. Sejnowski. Edges are the ‘independent components’ of natural scenes. In *Advances in Neural Information Processing Systems 9*, 1996.

[5] J.F. Cardoso. Infomax and maximum likelihood for blind source separation. *IEEE Letters on Signal Processing*, 4(4):112–114, 1997.

[6] G. Deco and L. Parra. Non-linear feature extraction by redundancy reduction in an unsupervised stochastic neural network. *Neural Networks*, 10(4):683–91, 1997.

[7] P. Földiák. Forming sparse representations by local anti-hebbian learning. *Biol. Cybern.*, 64:165–170, 1990.

[8] S. Haykin. *Neural Networks. A Comprehensive Foundation*. Macmillan College Publishing Company, 1994.

[9] J. Hertz, A. Krogh, and R.G. Palmer. *Introduction to the theory of neural computation*. Addison Wesley, 1991.

[10] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat’s visual cortex. *J.Physiol.*, 160:106–52, 1962.

[11] B.A. Olshausen. Learning linear, sparse, factorial codes. A.I. Memo 1580, MIT, 1996.

[12] B.A. Olshausen and D.J. Field. Sparse coding with an overcomplete basis set: A strategy employed by V1? *Vision Research*, 37:3311–3325, 1997.

[13] D. M. J. Tax and H. J. Kappen. Learning structure with many-take-all networks. In C. von der Malsburg, W. von Seelen, J. C. Vorbruggen, and B. Sendhoff, editors, *Proceedings ICANN*, pages 95–100, 1996.