Recurrent Sampling Models for the Helmholtz Machine

Peter Dayan

Department of Brain and Cognitive Sciences
E25-210, MIT
Cambridge, MA 02139

October 7, 1997

Abstract

Many recent analysis-by-synthesis density estimation models of cortical learning and processing have made the crucial simplifying assumption that units within a single layer are mutually independent given the states of units in the layer below or the layer above. In this paper, we suggest using either a Markov random field or an alternative stochastic sampling architecture to capture explicitly particular forms of dependence within each layer. We develop the architecture in the context of real and binary Helmholtz machines. Recurrent sampling can be used to capture correlations within layers in the generative or the recognition models, and we also show how these can be combined.

1 Introduction

Hierarchical probabilistic generative models have recently become popular for density estimation (Mumford, 1994; Hinton & Zemel, 1994; Zemel, 1994; Hinton *et al*, 1995; Dayan *et al*, 1995; Saul *et al*, 1995; Olshausen & Field, 1996; Rao & Ballard, 1997; Hinton & Ghahramani, 1997). They are statistically sound versions of a variety of popular unsupervised learning techniques (Hinton & Zemel, 1994), and they are also natural targets for much of the sophisticated theory that has recently been developed for tractable approximations to learning and inference in belief networks (Saul *et al*, 1995; Jaakkola, 1997; Saul & Jordan, 1997). Hierarchical models are also attractive for capturing cortical processing, finally giving some computational purpose to the top-down weights between processing areas which ubiquitously follow the rather better-studied bottom-up weights.

To fix the notation, figure 1 shows an example of a two-layer belief network which parameterises a probability distribution $\mathcal{P}[\mathbf{x}]$ over a set of activities of input units $\mathbf{x}$ as the marginal of the generative model $\mathcal{P}[\mathbf{x},\mathbf{y};\mathcal{G}]$:

$$\mathcal{P}[\mathbf{x};\mathcal{G}] = \sum_{\mathbf{y}} \mathcal{P}[\mathbf{x},\mathbf{y};\mathcal{G}]$$

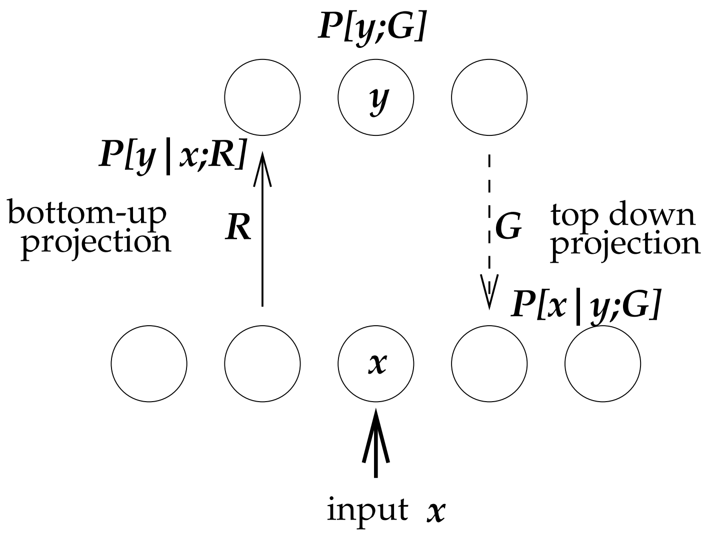

Figure 1: One-layer, top-down, generative model which specifies $\mathcal{P}[\mathbf{y};\mathcal{G}]$ and $\mathcal{P}[\mathbf{x}|\mathbf{y};\mathcal{G}]$ with generative weights $\mathcal{G}$. The recognition model specifies $\mathcal{P}[\mathbf{y}|\mathbf{x}]$ – the figure shows the Helmholtz machine version of this in which this distribution has parameters $\mathcal{R}$.

---

where $\mathbf{y}$ are the activities of the coding or interpretative units and $\mathcal{G}$ consists of all the generative parameters in the network. If $\mathbf{y}$ are real-valued, then the sum is replaced by an integral.

One constant facet of most of these generative models is that the units are organised into layers, and there are no connections between units within a layer, so that:

$$\mathcal{P}[\mathbf{y};\mathcal{G}] = \prod_j \mathcal{P}[y_j;\mathcal{G}] \tag{1}$$

$$\mathcal{P}[\mathbf{x}|\mathbf{y};\mathcal{G}] = \prod_i \mathcal{P}[x_i|\mathbf{y};\mathcal{G}] \tag{2}$$

This makes the x_i conditionally *factorial*, that is independent of each other given $\mathbf{y}$. The consequences of equations 1 and 2 are that the generative probability $\mathcal{P}[\mathbf{x},\mathbf{y};\mathcal{G}]$, given a complete assignment of $\mathbf{x}$ and $\mathbf{y}$ is extremely easy to evaluate, and it is also easy to generate a sample from the generative model. The Helmholtz machine (Hinton *et al*, 1995; Dayan *et al*, 1995) uses bottom-up weights to parameterise a *recognition* model, which is intended to be the statistical inverse to the generative model. It uses parameters $\mathcal{R}$ to approximate $\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}]$ by a distribution $\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}]$. One typical approximation in $\mathcal{Q}$ is that the units in $\mathbf{y}$ are *also* treated as being conditionally factorial, *ie* independent *given* $\mathbf{x}$.

Although these factorial assumptions are computationally convenient, there are various reasons to think that they are too restrictive. Saul & Jordan (1997) describe a convenient example from a generative standpoint. They built a hierarchical generative model that learns to generate 10×10 binary images of handwritten digits. However, the patterns that even a well-trained network tends to generate are too noisy – Saul & Jordan (1997) cite their network for lacking the means to perform 'clean-up' at the output layer. Clean-up is a characteristic role for Markov random fields in computer vision (*eg* Geman & Geman, 1984; Poggio, Gamble & Little, 1988), and is a natural task for lateral interactions. Equally, such lateral interactions can be used to create topographic maps, by encouraging neighbouring units to be correlated (Ghahramani & Hinton, 1998; Zemel & Hinton, 1995).

Equally, even for a generative model such as that in equations 1 and 2 in which the units $\mathbf{y}$ are unconditionally independent, once the values of $\mathbf{x}$ are observed, the $\mathbf{y}$ become dependent. This fact lies at the heart of the belief network phenomenon called *explaining away* (Pearl, 1988). In the simplest case of explaining away, two binary stochastic units y_a and y_b are unconditionally independent, and are both unlikely to turn on. However, if one (or both) of them turns on, then binary x is sure to come on too. This means that given the occurrence of $x = 1$, the recognition probability distribution over $\{y_a, y_b\}$ should put its weight on $\{1, 0\}$ and $\{0, 1\}$, and not on $\{0, 0\}$ (since *something* has to explain $x = 1$) or $\{1, 1\}$ (since y_a and y_b are individually unlikely). Therefore, the presence of $y_a = 1$ explains away the need for $y_b = 1$ (and vice-versa). Modeling this conditional dependence requires something like a large and negative lateral recognition influence between y_a and y_b. Therefore, it is inadequate to model the recognition distribution $\mathcal{Q}[\mathbf{y}; \mathbf{x}, \mathcal{R}]$ as being factorial.

Finally, although all the statistical models are still much too simplistic, we are interested in using them to capture aspects of cortical processing and learning. As has long been known from work in Markov random fields, statistical models whose units have lateral connections (*ie* undirected connections and loops) require substantially different treatment from statistical models without lateral connections. It is worthwhile exploring even simple members of the latter class, since lateral connections are ubiquitous in the cortex.

Even at a coarse level of descriptive detail, there are at least two different classes of lateral connections, and we might expect these to have distinctive roles. One set comprises the connections that form the intra-columnar canonical microcircuit (see Douglas, Martin & Whitteridge, 1989; Douglas & Martin, 1990;1991), connecting cells in layer IV, which receive inputs from lower layers in the hierarchy, to cells in layer II/III (these are sometimes called vertical connections). The other set is inter-columnar, connecting cells in layers II/III (see Gilbert, 1993; Levitt, Lund & Yoshioka, 1996; Fitzpatrick, 1996). The latter class (also called horizontal) are more conventionally thought of as being the lateral connections.

Some hints come from the course of development as to the nature of these connections. For instance, in humans, top-down connections from area V2 to area V1 innervate V1 from birth. However, these fibres terminate in the lowest layers of the cortex, making just a few synaptic contacts until around three months. At this point they grow up through the cortical layers and form what are the majority of their synapses, onto cells in layers II/III. At just this same juncture, axons from other layer II/III neurons in V1 are also growing and making contacts (Burkhalter, 1993). This suggests the possibility that top-down and lateral connections might play similar roles, putatively both being part of the generative model.

We therefore seek ways of using lateral interactions to represent dependencies between units within a layer. The issues are how lateral *weights* can parameterise statistically meaningful lateral interactions, how they can be learned, and, crucially, how it is possible to capture both generative *and* recognition dependencies in one network. The Boltzmann machine (BM; Hinton & Sejnowski, 1986) is a very natural model for lateral connections, and sampling and learning in such Markov random fields is quite well understood. We consider the BM, and also a different recurrent sampling model that obviates the need for the BM's negative sampling phase (also called the sleep phase). We will focus on two models. One is the simplest linear and Gaussian generative model, which performs the statistical technique of factor analysis (see Everitt, 1984), since this is a paradigmatic example of the Helmholtz machine (Neal & Dayan, 1997), and since a mean field version of it has been extensively investigated (Rao & Ballard, 1997). However, lateral models are more interesting in non-Gaussian cases, an extreme example of which involves just binary activations of the units, and we discuss and experimentally investigate this too.

2 Factor Analysis

Consider the special case of figure 1 and equation 2 in which the units are linear and the distributions are Gaussian:

$$\mathbf{y} \sim \mathcal{N}[\mathbf{0}, \Phi] \tag{3}$$

$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}\left[\mathsf{G}^T\mathbf{y}, \Psi\right] \tag{4}$$

$$\Psi = \text{diag}\left(\tau_1^2, \ldots, \tau_n^2\right) \tag{5}$$

where $\mathcal{N}[\boldsymbol{\mu}, \Gamma]$ is a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix Γ.[1] We have omitted the bias terms for convenience. This is just the standard factor analysis model in statistics (*eg* Everitt, 1984). If Φ is a multiple of the identity matrix, I, then the $\mathbf{y}$ are unconditionally independent (and therefore satisfy equation 1); otherwise they are unconditionally dependent. The task for maximum likelihood factor analysis is to take a set of observed patterns $\{\mathbf{x}^\bullet\}$ and fit the parameters $\mathcal{G}$ of the model to maximise their likelihood.

[1] Note the different symbols: $\mathcal{G}$ is the entire set of generative parameters, including the weights G and the variances $\{\tau_i^2\}$. For the moment, we treat the covariance matrix Φ as being fixed.

The recognition model is just the statistical inverse of the generative model in equation 5. In this case, $\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}]$ is also Gaussian:

$$\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}] \sim \mathcal{N}[\mathsf{R}^{*T}\mathbf{x}, \Sigma^*], \qquad \text{where} \tag{6}$$

$$\mathsf{R}^* = \Psi^{-1}\mathsf{G}\left(\Phi^{-1} + \mathsf{G}\Psi^{-1}\mathsf{G}^T\right)^{-1} \tag{7}$$

$$\Sigma^{*-1} = \Phi^{-1} + \mathsf{G}\Psi^{-1}\mathsf{G}^T \tag{8}$$

Note that the mean of $\mathbf{y}|\mathbf{x}$ depends just linearly on the input $\mathbf{x}$, and the covariance matrix Σ^* does not depend on $\mathbf{x}$ at all. The covariance matrix captures the conditional dependence amongst the $\mathbf{y}$ during recognition.

One standard way of performing factor analysis is the expectation maximisation algorithm (EM; Dempster *et al*, 1977; Rubin & Thayer, 1982). During the E phase, an input $\mathbf{x}$ is presented, and the distribution $\mathcal{P}[\mathbf{y}|\mathbf{x}]$ is determined. During the M phase, the generative weights (G and Ψ here) are updated in the light of $\mathbf{x}$ and $\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}]$. Neal & Dayan (1997) showed that the wake-sleep algorithm (Hinton *et al*, 1995) can be used to learn the generative and recognition weights. Wake-sleep is an iterative approximate form of EM, which explicitly maintains parameters $\mathcal{R}$ for a current recognition model $\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}]$, and requires for learning nothing more than two phases of application of the delta rule. During the wake phase, patterns $\mathbf{x}^\bullet$ are drawn from their distribution in the environment, and a sample $\mathbf{y}^\bullet$ is drawn using the current recognition distribution, $\mathcal{Q}[\mathbf{y}^\bullet;\mathbf{x}^\bullet,\mathcal{R}]$. Then the delta rule is used to adapt G and $\{\tau_i^2\}$ to reduce $\left(\mathbf{x}^\bullet - \mathcal{G}^T\mathbf{y}^\bullet\right)^T \Psi^{-1} \left(\mathbf{x}^\bullet - \mathcal{G}^T\mathbf{y}^\bullet\right) + \sum_i \log\tau_i^2$. During the sleep phase, samples $\mathbf{y}^\circ, \mathbf{x}^\circ$ are drawn top-down from the generative model, and the parameters of the recognition model are changed using the delta rule again to decrease $-\log\mathcal{Q}[\mathbf{y}^\circ|\mathbf{x}^\circ;\mathcal{R}]$. The obvious parameterisation to use for $\mathcal{Q}$ is $\mathcal{R} = \{\mathsf{R},\Sigma\}$, where $\mathcal{R}$ is an approximation of $\mathcal{R}^*$ and Σ an approximation of Σ^*.

We have therefore specified both generative (Φ) and recognition (Σ) covariances. In this paper, we focus on their representation and acquisition in lateral connections. Neal & Dayan (1997) suggested two possibilities for representing Σ. One is to note that if the generative prior over $\mathbf{y}$ is rotationally invariant, so Φ is a multiple of I (which itself is easy to represent), then there is rotational redundancy in the definition of $\mathbf{y}$ and G. This means that both can be multiplied appropriately by any unitary matrix without affecting the underlying generative model. In particular, there will always be one privileged rotation in which the recognition covariance matrix Σ will be diagonal. The diagonal terms are straightforward to learn, again using the delta rule. In tests, this model worked quite well, but would occasionally get stuck in a local minimum.

In the more interesting case in which Φ is not completely rotationally invariant, it may not be possible to capture general Σ using only those rotations that are permissible. The other suggestion in Neal & Dayan (1997) was to connect the units in $\mathbf{y}$ with the ladder (Markov mesh) structure shown in figure 2 (see also Frey *et al*, 1995; Frey, 1997). This has just enough

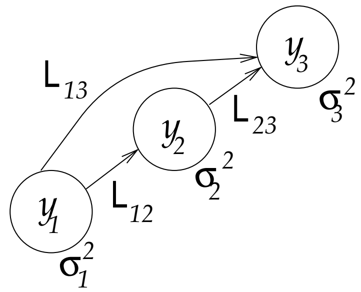

Figure 2: The ladder of connections L and the individual covariance terms σ_i^2 that are required to capture a full recognition covariance matrix.

---

representational capacity to model an arbitrary full covariance matrix Σ, and can also be learnt using the standard delta rule. However, the requirement that the connections be laddered is rather inelegant. In non-Gaussian cases, arbitrary dependencies can be captured by laddered models only if the unit activation functions are allowed to be sufficiently complex. For a given activation function, it can be that some dependence is overlooked, and that omitting half the connections is harmful. Further, if Φ is not rotationally invariant, then these generative covariances need to be represented too.

It is therefore natural to seek a model which can represent arbitrary Φ and Σ with fully bidirectional (but not necessarily symmetric-valued) connections, and can also learn appropriate values for these connections.

3 Lateral Models

One important facet of the wake-sleep algorithm is that the activities of the hidden units are specified by the recognition model whilst the generative model is plastic, and *vice-versa.* This implies that it is unnecessary to extract samples from a model when its weights are actually being changed. Consider the case of representing and learning the generative covariances Φ. The task becomes, given samples $\mathbf{y}^\bullet$ (which are produced by the recognition model), specify an architecture and learning scheme such that arbitrary new samples can be drawn from the distribution $\mathcal{P}[\mathbf{y}^\bullet]$ during the sleep phase.

The Gaussian Boltzmann Machine

An obvious choice for a lateral model for the case of factor analysis is a Gaussian-valued Boltzmann machine. In this, one would have an energy function defined as:

$$E[\mathbf{y}] = -\frac{1}{2}\mathbf{y}^T \mathsf{W}\mathbf{y}$$

with a symmetric, negative definite matrix W with $\mathsf{W}_{ii} < 0$ and $\mathsf{W}_{ij} = \mathsf{W}_{ji}$.[2] This energy function is used to define a probability distribution according to

$$\mathcal{P}[\mathbf{y}] = e^{-E[\mathbf{y}]}/\mathcal{Z}[\mathsf{W}] \tag{9}$$

where $\mathcal{Z}[\mathsf{W}]$ is the partition function ($\int_{\mathbf{y}} e^{-E[\mathbf{y}]}d\mathbf{y} = (2\pi)^{n/2}/\sqrt{|-\mathsf{W}|}$, where n is the dimensionality of $\mathbf{y}$). Clearly, equation 9 is just a Gaussian distribution, with covariance matrix $-\mathsf{W}^{-1}$. We would therefore like W to come to equal $-\Phi^{-1}$.

Given W, we can extract samples from the distribution using the Markov chain Monte-Carlo method called Gibbs sampling (see Neal, 1994 for an excellent review of Markov chain methods). For this, we sample y_i from the distribution defined by all the other $y_j, j \neq i$ (which we will call $\mathbf{y}_{\bar{\imath}}$). This is the Gaussian:

$$\mathcal{P}[y_i|\mathbf{y}_{\bar{\imath}}] = \mathcal{N}\left[\frac{-1}{\mathsf{W}_{ii}}\sum_{j\neq i}\mathsf{W}_{ij}y_j, \frac{-1}{\mathsf{W}_{ii}}\right] \tag{10}$$

Provided that we choose the order of updates appropriately, then Gibbs sampling is a natural (albeit possibly slow) way by which to express the distribution. The standard alternative method involves diagonalising Φ, which is less practicable using local operations.

The next issue for the Gaussian BM concerns learning, which, at least traditionally, involves two phases (both of which happen during the wake phase of Helmholtz machine). Of course, learning Φ is *trivial,* since one only needs to observe the correlations $\langle \mathbf{y}_i^\bullet \mathbf{y}_j^\bullet \rangle$ where $\mathbf{y}^\bullet$ are samples provided by the recognition model. Learning Φ^{-1}, which is required to be able to produce samples during the sleep phase, is more complicated.

Positive learning for the BM with fully specified samples is again easy:

$$\Delta\mathsf{W}_{ij}^{+} \propto \nabla_{\mathsf{W}_{ij}}\left\langle -E\left[\mathbf{y}^\bullet\right]\right\rangle \propto \left\langle \mathbf{y}_i^\bullet \mathbf{y}_j^\bullet \right\rangle$$

[2]The choice of W rather than $-\mathsf{W}$ is somewhat arbitrary – whereas a standard Gaussian has a convex energy function, the standard Boltzmann machine does not (it typically consists of multidimensional saddles). If $\mathsf{W}_{ij} \gg 0$, then the energy is low (and consequently the probability is high) if y_i and y_j are large and have the same sign.

where, $\mathbf{y}^\bullet$ are once more samples provided by the recognition model. The negative phase of BM learning is not so easy. Since the partition function is analytically calculable, we know that

$$\Delta \mathsf{W}^-_{ij} \propto \nabla_{\mathsf{W}_{ij}} \log \mathcal{Z}[\mathsf{W}] \propto \nabla_{\mathsf{W}_{ij}} - \log|-\mathsf{W}| = -\mathsf{W}^{-1}_{ij}$$

(since $\mathsf{W} = \mathsf{W}^T$). Since equation 9 really expresses a multivariate Gaussian distribution with covariance matrix $-\mathsf{W}^{-1}$, one could estimate $-\mathsf{W}^{-1}_{ij} = \langle \mathbf{y}^\dagger_i \mathbf{y}^\dagger_j \rangle$ using Gibbs sampling according to equation 10 to produce samples $\mathbf{y}^\dagger$ from equation 9. It is the closed form for $\mathcal{Z}[\mathsf{W}]$ that makes it unnecessary.

Combining the two contributions to the weight change, this would make:

$$\Delta \mathsf{W}_{ij} = \Delta \mathsf{W}^+_{ij} - \Delta \mathsf{W}^-_{ij} \propto \left\langle \mathbf{y}^\bullet_i \mathbf{y}^\bullet_j \right\rangle + \mathsf{W}^{-1}_{ij}$$

Just as in Amari (1997), the requirement for inverting W can be averted through multiplying this learning rule by $\mathsf{W}\mathsf{W}^T = \mathsf{W}\mathsf{W}$, giving

$$\Delta \mathsf{W} \propto \mathsf{W}\mathsf{W} \left\langle \mathbf{y}^\bullet \mathbf{y}^{\bullet T} \right\rangle + \mathsf{W}.$$

In this Gaussian case, it is therefore possible to avoid the BM's normal requirement for a negative phase of learning, since the partition function is analytically calculable. This simplification is not available for the case of stochastic binary units.

The Direct Method

There is an alternative to using the Gaussian Boltzmann machine. Equation 10 specifies as a Gaussian the conditional distribution of y_i given all the other $\mathbf{y}_{\bar{\imath}}$. The mean of this Gaussian depends linearly on these other variables and its variance is independent of them. Imagine just learning the parameters of these conditional distributions – *ie* learning V and $\phi_i^2 = e^{\beta_i}$ where

$$\mathcal{P}[y_i | \mathbf{y}_{\bar{\imath}}; \mathsf{V}] = \mathcal{N}\left[\sum_{j \neq i} \mathsf{V}_{ij} y_j, \phi_i^2\right] \tag{11}$$

using the delta rule:

$$\begin{aligned}
\Delta \mathsf{V}_{ik} &\propto \frac{1}{\phi_i^2}\left(y^\bullet_i - \textstyle\sum_{j\neq i} \mathsf{V}_{ij} y^\bullet_j\right) y^\bullet_k \\
\Delta \beta_i &\propto \frac{1}{\phi_i^2}\left(\left(y^\bullet_i - \textstyle\sum_{j\neq i} \mathsf{V}_{ij} y^\bullet_j\right)^2 - \phi_i^2\right).
\end{aligned}$$

based on samples $\mathbf{y}^\bullet$ drawn from the recognition model. The delta rule is perfectly local, and is exactly the learning rule used for all other parts of the Helmholtz machine. In this linear case, when applied with suitable schedule for changing the learning rates, the delta rule is provably a convergent way of determining $\mathcal{P}[y_i|\mathbf{y}_{\bar{\imath}}]$. Therefore, the rule will ultimately find appropriate lateral weights. This is again without requiring a negative phase of learning, and also without requiring an analytical form for the partition function.

However, this rule is quite different in form from the BM learning rule. For instance, note that in general, $\mathsf{V}_{ij} \neq \mathsf{V}_{ji}$. Potentially more worrying, for intermediate values of V and ϕ_i^2 before convergence, the sampler (a stochastic cellular automaton, see, *eg*, Marroquin & Ramirez, 1991) defined by equation 11 is not nearly as well behaved as that defined by equation 10. As an example, for equation 10, the precise details of the way that the order of update of the $\{y_i\}$ is irrelevant, provided that all the states are updated sufficiently often. This is *not* true for equation 11, since there can be the stochastic equivalent of cycles. For instance, consider the case in which there is just y_1 and y_2, which are updated sequentially according to

$$\begin{array}{lrcl} u_1) & y_2' & = & by_1 + \epsilon_2 \\ u_2) & y_1' & = & ay_2' + \epsilon_1 \end{array}$$

where ϵ_i are Gaussian random variables (distributed according to $\mathcal{N}[0,1]$) and a and b are weights. If this has well-defined terminal behavior (we will see later circumstances in which it may not) then we can ask whether the distribution of $\{y_1, y_2\}$ depends on whether we stop to take samples after update u_1 or update u_2. In fact, the distribution does indeed depend on this – the asymptotic covariance matrices of the samples would be

$$\Xi^{u_1} = \frac{1}{1-a^2b^2}\begin{pmatrix} 1+a^2 & a(1+b^2) \\ a(1+b^2) & 1+b^2 \end{pmatrix} \quad \Xi^{u_2} = \frac{1}{1-a^2b^2}\begin{pmatrix} 1+a^2 & b(1+a^2) \\ b(1+a^2) & 1+b^2 \end{pmatrix}$$

and so only if $a = b$ (or the degenerate case of $ab = 1$) are these the same. Of course, at the point of convergence of learning, since equations 10 and 11 are the same, then the order ceases to matter. Also, one could artificially force the connections to be symmetric by averaging the weight changes in both directions.

Provided the update order is consistent (or consistently random) this might not matter. Worse is the possibility that the iteration in equation 11 is divergent. This arises from the fact that making the individual conditional probabilities closer to being correct does not have a provable relationship to making correct the stationary distribution defined by the full Markov chain Monte-Carlo method. For the simple example above, if $ab > 1$, then the magnitudes of y_1 and y_2 will get ever larger and the iteration will not lead to a well defined terminal distribution. This never happened in empirical investigations, and is in any case avoided in non-linear cases with saturation such as stochastic binary units.

We have therefore defined two ways of allowing for a full generative covariance matrix for Gaussian factor analysis, at the expense of having to use a Markov chain Monte-Carlo technique to generate samples. One of the methods is based on the Gaussian BM. The positive phase of the BM is in any case easy, since there are no hidden units. The negative phase was made redundant by virtue of the exact partition function and the natural gradient trick of Amari (1997). The other method, which we call the 'Direct' method, abandoned the energy function of the BM, and instead set out to learn a sampler directly. This has some attractive features, although one cannot rule out *a priori* the possibility that at some intermediate point of learning, the resulting sampler may not work.

The Recognition Model

Exactly the same architecture and learning could have been used to learn the recognition model instead of the generative model. In this case, samples $\mathbf{x}^\circ$ and $\mathbf{y}^\circ$ are drawn from the generative model during the sleep phase of the HM, there are feedforward recognition weights R from $\mathbf{x}$ to $\mathbf{y}$, lateral weights V and variances $\phi_i^2 = e^{\beta_i}$ within the $\mathbf{y}$ layer, and a Gaussian sampling distribution:

$$\mathcal{P}[y_i|\mathbf{y}_{\bar{\imath}}, \mathbf{x}] \sim \mathcal{N}\left[\left[\mathsf{R}^T\mathbf{x}\right]_i + \sum_{j\neq i}\mathsf{V}_{ij}y_j, \phi_i^2\right] \tag{12}$$

The weights can be learnt using exactly the delta rule that is used for wake-sleep:

$$\Delta\mathsf{R}_{ki} \propto \frac{1}{\phi_i^2}\left(y_i^\circ - \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i - \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)x_k^\circ \tag{13}$$

$$\Delta\mathsf{V}_{ik} \propto \frac{1}{\phi_i^2}\left(y_i^\circ - \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i - \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)y_k^\circ \tag{14}$$

$$\Delta\beta_i \propto \frac{1}{\phi_i^2}\left(\left(y_i^\circ - \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i - \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)^2 - \phi_i^2\right). \tag{15}$$

This again involves no sampling during learning and nothing like the negative phase of the BM.

However, there is an alternative way of implementing the recognition model that fits better with a putative mapping onto cortex. Consider the case in equations 3, 4 and 5. Write $\lambda_i^y = \left[\mathsf{G}\Psi^{-1}\mathsf{G}\right]_{ii}$, $\mu_i^y = \Phi_{ii}^{-1}$, and $\mu_i^x = \Psi_{ii}^{-1}$, and consider the direct method version in which $\mathsf{V}_{ij}^y = -\Phi_{ij}^{-1}/\mu_i^y$ and $\mathsf{V}_{ij}^x = -\Psi_{ij}^{-1}/\mu_i^x$. Then, one can write:

$$\mathcal{P}[y_i|\mathbf{x}, \mathbf{y}_{\bar{\imath}}] \sim \mathcal{N}\left[\frac{1}{\mu_i^y + \lambda_i^y}\left(\mu_i^y\sum_{j\neq i}\mathsf{V}_{ij}^y y_j + \left[\mathsf{G}\Psi^{-1}(\mathbf{x} - \mathsf{G}^T\mathbf{y})\right]_i + \lambda_i^y y_i\right), \frac{1}{\mu_i^y + \lambda_i^y}.\right] \tag{16}$$

There is an extra constraint that is important for a truly hierarchical system. Consider unit a in the $\mathbf{x}$ layer. This unit gathers information from the other units in the $\mathbf{x}$ layer to sample its state, but also has to feed information to unit in the $\mathbf{y}$ layer as part of the information they use to sample their states. It is important that the difference between what it receives and what it sends should be calculable using just the information this unit possesses. This is in fact true here, since unit a in the $\mathbf{x}$ layer receives

$$\gamma_a = \sum_{b \neq a} V^x_{ab} \left[\mathbf{x} - \mathsf{G}^T \mathbf{y} \right]_b$$

from its neighbours in the $\mathbf{x}$ layer to determine its sampling, and need only broadcast the difference between its actual value and this mean:

$$\left[\Psi^{-1} (\mathbf{x} - \mathsf{G}^T \mathbf{y}) \right]_a = \mu^x_a \left(x_a - \left[\mathsf{G}^T \mathbf{y} \right]_a - \gamma_a \right)$$

up to the $\mathbf{y}$ layer.

There are two ways to treat the expression in equation 16. The first is exactly as in equations 13, 14 and 15. In this case, one would learn a set of weights $\left[\mathsf{G} \mathsf{V}^x \mathsf{G}^T \right]_{ij}$ between units i and j in the y layer which are in *addition* to the weights V^y weights that define the generative model. One would also use as bottom-up weights from the $\mathbf{x}$ layer essentially the transpose of the generative weights. Hinton & Ghahramani (1997) suggest a close analogue of this for their rectified Gaussian belief nets, and suggest exactly how these bottom-up and lateral weights could be learnt. Unless symmetry in the weights is explicitly enforced, the resulting architecture at any intermediate state of learning must be analysed as an example of the direct method rather than a BM.

The second way to treat equation 16 is to define dynamics within the $\mathbf{x}$ layer such that the difference between the actual activities and the top-down predictions of those activities (*ie* $\mathbf{x} - \mathsf{G}^T \mathbf{y}$) is propagated bottom-up. Rao & Ballard (1997) use this effect to model various properties of cortical representations, and suggest how the required bottom-up weights $\mathsf{G}^T \Psi^{-1}$ could be learnt. It is then necessary to learn λ^y_i, which is used as a weighting factor which determines the relative influence of top-down and bottom-up connections during the phase of recognition sampling.

Even though equation 16 suggests how to perform stochastic sampling, both these ways of handling explaining away have emerged in various deterministic mean field algorithms (Jaakkola *et al*, 1996; Rao & Ballard, 1997; Olshausen & Field, 1996; Dayan, 1997). In the terms of this paper, Rao & Ballard (1997) suggest finding the representation $\mathbf{y}$ for a particular $\mathbf{x}$ by minimising an expression:

$$E[\mathbf{y}] = \frac{1}{2} \left(\left(\mathbf{x} - \mathsf{G}^T \mathbf{y} \right)^T \Psi^{-1} \left(\mathbf{x} - \mathsf{G}^T \mathbf{y} \right) + \mathbf{y}^T \Phi^{-1} \mathbf{y} \right)$$

which, up to some constant factors, is exactly the negative log-likelihood under the factor analysis model. Olshausen & Field (1996) pointed out that there are two obvious iterative gradient descent algorithms for doing this:

$$\dot{\mathbf{y}} = -\Phi^{-1}\mathbf{y} + \mathsf{G}\Psi^{-1}\mathbf{x} - \left(\mathsf{G}\Psi^{-1}\mathsf{G}^T\right)\mathbf{y} \tag{17}$$

$$\dot{\mathbf{y}} = -\Psi^{-1}\mathbf{y} + \mathsf{G}\Psi^{-1}\left(\mathbf{x} - \mathsf{G}^T\mathbf{y}\right). \tag{18}$$

Both iterations use the transpose of the top-down weights as bottom-up weights. Equation 17 uses additional lateral connections $-\left(\mathsf{G}\Psi^{-1}\mathsf{G}^T\right)$ between the $\mathbf{y}$ units; equation 18 uses the dynamics in the $\mathbf{x}$ layer. Of course, in this simple Gaussian case, it is not necessary to perform either iteration to find the true mean $\mathbf{y}$. Rather, this can be accomplished in a single bottom-up step using the weights given in equation 7, although integrating bottom-up and top-down information correctly will require iteration.

These mean field methods just find the *mode* of the distribution (which, because of its Gaussian form, is also the mean). However, we see from equation 16, that having the capacity to *sample* from the correct full distribution, including the covariance, requires the same information. The only difference is that the influence of y_i itself has to be subtracted out according to a constant factor λ_i that, crucially, does not depend on the value of the inputs $\mathbf{x}$. For the Gaussian model, the deterministic and the stochastic models are extremely close. By reducing the variance of the added noise in equation 16 away from its normative value, one could move smoothly between slower, sampled, but statistically correct recognition and faster, deterministic, but mean-field recognition.

Note that there is a difference between the correct bottom-up weights in equation 7 which are intended for bottom-up inference in the absence of information about the activities of the other $\mathbf{y}_{\bar{\imath}}$, and the bottom-up weights ($\mathsf{G}\Psi^{-1}$) in the iterative sampling scheme in equation 16. The difference is the 'shrinkage' factor $\Phi^{-1} + \mathsf{G}\Psi^{-1}\mathsf{G}^T$. This arises since, if there is to be no repeated sampling, then the bottom-up weights have to take account of the prior over $\mathbf{y}$; whereas if there is repeated sampling, then this prior is taken account of directly. For instance, if $\Phi = \epsilon\mathsf{I}$ for some very small ϵ, then the mean value of $\mathbf{y}$ given $\mathbf{x}$ will also be quite small. If bottom-up weights from $\mathbf{x}$ are used as in equation 7, then they will have small magnitudes; if an iterative scheme is used instead, then this is captured in the multiplication factor $1/(\Phi_{ii}^{-1} + \lambda_i)$ for the mean.

There is a final implementational possibility that integrates the generative and recognition versions and provides a map onto cortical microcircuitry (see also Rao & Ballard, 1997). If the bottom-up weights specify R^* as in equation 7, then they can be used to provide a fast, first, estimate of the mean into layer IV. These can be learnt using standard wake-sleep learning. If top-down information has to be integrated too, then the weights from layer IV to layers II/III can implement the matrix $\Phi^{-1} + \mathsf{G}\Psi^{-1}\mathsf{G}^T$ (which can be learnt using the delta rule as in the direct method, provided that λ_i^y is taken into proper account), and the iterative sampling scheme using the dynamics in the $\mathbf{x}$ layer will perform true Gibbs sampling for the full model. Switching between a bottom-up mode and a mode with a mixture of bottom-up, top-down and lateral influences would require two things. One is a multiplicative modulation of excitability. The other is a mild readjustment in the nature of the connections from layer IV to layer II/III, in that instead of having layer II/III cells follow the layer IV input faithfully, they should integrate all sources of information, even including a wider range of layer IV inputs. Hasselmo (1995) and Hasselmo & Cekic (1996) report effects for neuromodulators such as acetylcholine on the cortex (and the hippocampus), and Hasselmo (personal communication) has suggested something similar for GABA_B receptors which can be loosely interpreted in this light, although the experimental results are not nearly as specific as this theory would require.

4 The Binary Case

We can also consider the direct method in the case of the binary stochastic belief net that was the original target of the wake-sleep algorithm and the Helmholtz machine. In the simple case of figure 1, this has for equations 1 and 2:

$$\mathcal{P}[\mathbf{y};\mathcal{G}] = \prod_j \rho(b_j)^{y_j}\rho(-b_j)^{1-y_j} \tag{19}$$

$$\mathcal{P}[\mathbf{x}|\mathbf{y};\mathcal{G}] = \prod_i \rho\left(\left[\mathsf{G}^T\mathbf{y}\right]_i\right)^{x_i}\rho\left(-\left[\mathsf{G}^T\mathbf{y}\right]_i\right)^{1-x_i} \tag{20}$$

where $\rho(a) = 1/(1+e^{-a})$ is the standard sigmoid function, and $\mathbf{b}$ are the biases for the activities of $\mathbf{y}$.

We will consider using lateral connections in the recognition model. In this case, there is no such convenient representation for the true recognition distribution as equation 6. In the Helmholtz machine, we attempted to learn a factorial model:

$$\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}] = \prod_j \rho\left(\left[\mathsf{R}^T\mathbf{x}\right]_j\right)^{y_j}\rho\left(-\left[\mathsf{R}^T\mathbf{x}\right]_j\right)^{1-x_j} \tag{21}$$

even though, in cases such as explaining away, the true distribution of $\mathbf{y}$ given $\mathbf{x}$ is not factorial. The effect of this lack of expressive power is made more severe in the wake-sleep algorithm by the fact that the learning rule during sleep is based on the 'wrong' Kullback-Leibler divergence. Rather than choosing $\mathcal{R}$ to minimise an expression equivalent to

$$KL[\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}],\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}]] = \sum_{\mathbf{y}} \mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}]\log\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}]/\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}],$$

sleep learning minimises

$$KL[\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}],\mathcal{Q}[\mathbf{y};\mathbf{x},\mathcal{R}]],$$

and, in the case that it is impossible to get to $\mathcal{P}[\mathbf{y}|\mathbf{x};\mathcal{G}] = \mathcal{Q}[\mathbf{y};\mathbf{x};\mathcal{R}]$ (which is the optimum point for both), minimising the two different Kullback-Leibler divergences can lead to two different answers.

In this case, it is again natural to express the dependence between y_a and y_b using a binary stochastic BM. Including the biases and the effect of the input $\mathbf{x}$, the energy function and associated probabilities are:

$$\begin{aligned} E[\mathbf{y}|\mathbf{x}] &= -\frac{1}{2}\sum_{ij} y_i \mathsf{W}_{ij} y_j - \sum_i y_i \left(b_i + \left[\mathsf{R}^T\mathbf{x}\right]_i\right) \\ \mathcal{P}[\mathbf{y}|\mathbf{x}] &= e^{-E[\mathbf{y}|\mathbf{x}]}/\mathcal{Z}[\mathsf{W},\mathbf{x}], \end{aligned}$$

where $\mathsf{W}_{ij} = \mathsf{W}_{ji}$ and $\mathsf{W}_{ii} = 0$, and $\mathcal{Z}[\mathsf{W},\mathbf{x}]$ is the partition function, which is a sum over the 2^n possible discrete binary states. In this case, $\mathcal{Z}[\mathsf{W},\mathbf{x}]$ *can* depend on $\mathbf{x}$. For the binary BM, the conditional distributions of y_i given $\mathbf{y}_{\bar{\imath}}$ and $\mathbf{x}$ that can be used for Gibbs sampling are:

$$\mathcal{P}\left[y_i = 1|\mathbf{y}_{\bar{\imath}}\right] = \rho\left(b_i + \left[\mathsf{R}^T\mathbf{x}\right]_i + \sum_{j\neq i}\mathsf{W}_{ij}y_j\right)$$

The trouble for the BM is that there is generally no closed form expression for the partition function. This leads directly to the requirement for the negative phase of learning.

The Direct method has exactly the same form as above. Now, the weights V directly parameterise the conditional probabilities for sampling,

$$\mathcal{P}\left[y_i = 1|\mathbf{y}_{\bar{\imath}},\mathbf{x}\right] = \rho\left(b_i + \left[\mathsf{R}^T\mathbf{x}\right]_i + \sum_{j\neq i}\mathsf{V}_{ij}y_j\right),$$

and learning again uses the delta rule:

$$\begin{aligned} \Delta b_i &\propto \left(y_i^\circ - \rho\left(b_i + \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i + \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)\right) \\ \Delta \mathsf{R}_{ki} &\propto \left(y_i^\circ - \rho\left(b_i + \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i + \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)\right)x_k^\circ \\ \Delta \mathsf{V}_{ik} &\propto \left(y_i^\circ - \rho\left(b_i + \left[\mathsf{R}^T\mathbf{x}^\circ\right]_i + \textstyle\sum_{j\neq i}\mathsf{V}_{ij}y_j^\circ\right)\right)y_k^\circ \end{aligned}$$

based on samples $\mathbf{x}^\circ$ and $\mathbf{y}^\circ$ from the process that truly generates the data. If this process happened to be a Boltzmann machine, then this method will learn to invert it exactly. If the generative process was not a BM, then it is not so clear to what it will converge. Again, making the conditional probabilities as close as possible (*ie* as close as the method can parameterise) to being correct does not necessarily make the stationary distribution for the overall Markov chain as close as the method can parameterise.

Unfortunately, because of the non-Gaussian nature of the probabilities, it is no longer possible to derive a sampling scheme such as that in equations 16 to combine top-down and bottom-up inference. True Gibbs sampling in this method requires significantly more complicated calculations whose neural instantiation is uncertain.

5 Comparisons

It is most instructive to study the Direct method in the case of binary rather than Gaussian units, since we can calculate the partition function for the BM in closed form in the Gaussian case. We performed two experiments – one which studies the two methods in isolation, and the second which uses them in the context of wake-sleep sampling and a hierarchical generative model.

Isolated Models

Figure 3 shows results comparing the BM with the Direct method for learning two sizes of BM. First, random weights were drawn from a uniform distribution in $[-3, 3]$, and the resulting BM used to generate a set of 5000 learning patterns. Then these patterns were fed either to a BM or to the Direct method. The proximity between the resulting model and the original BM was assessed by measuring the Kullback-Leibler distance between their distributions. For the BMs, this was assessed explicitly. For the Direct method, this was assessed by calculating empirically the stationary distribution of the stochastic automaton.

Since the Direct method avoids the negative phase of learning, we compared it both with a BM whose computational demands are equivalent (called BM(1) in the figure) and a more accurate implementation of the BM (called BM(64)). The difference between these two is the number of Gibbs sampling sweeps across all the units on each negative phase before taking a single learning sample. BM(1) only takes one sweep, and therefore the statistics of its learning sample are unlikely to be that close to that of the real underlying Boltzmann distribution. BM(64) takes 64 sweeps. Although its learning samples are undoubtedly better (as is confirmed by the fact that it learns faster), BM(64) pays a substantial computational cost, and still absorbs significantly less information from training examples than the Direct method. It is possible that the BM results could have been improved given an appropriate annealing schedule.

Figure 3: BM *versus* the direct method. The graphs show the average KL distance after 5000 learning samples between 100 target distributions over 6 (left) and 10 (right) units and the stationary distribution of a learned network. The target distributions were generated from BMs with random weights. ϵ is the learning rate in both cases. 'BM (1)' indicates that only one Gibbs sampling updates was used in the negative phase of BM learning before a learning sample was drawn; 'BM (64)' indicates that 64 Gibbs sampling updates were used.

Wake-Sleep Learning

Although these results favor the Direct method when run in isolation, it remains to be shown that the Direct method will actually work when embedded in the full context of wake-sleep. We therefore tried it on the bars problem that has been extensively used as a test case for unsupervised learning algorithms. For our version, 6×6 binary images contain either horizontal *or* vertical bars, but not both. Figure 4a shows some examples of the training patterns. The wake-sleep algorithm should infer that bars are hidden 'causes' of correlations in the activity of input units, and should therefore learn to represent new images of bars in their terms. It should also pick out the further regularity that horizontal and vertical bars do not co-occur. In earlier work on the bars problem (Hinton *et al*, 1995) we used a hierarchical generative model, in which a single unit in the top layer made the decision between horizontal and vertical bars (figure 4c(i)). However, this can equally well be done using connections between units within a single hidden layer, as in figure 4c(ii), in which the units representing all the horizontal bars inhibit the units representing the vertical bars, and vice-versa. We sought to learn such a lateral generative model using either the BM or the Direct method. We employed 15 hidden units in the $\mathbf{y}$ layer (which is 3 more than necessary).

Hinton *et al* (1995) arranged for the wake-sleep algorithm to work on a 4×4 version of the bars problem by forcing the generative weights from $\mathbf{y}$ to $\mathbf{x}$ to be positive and by using a

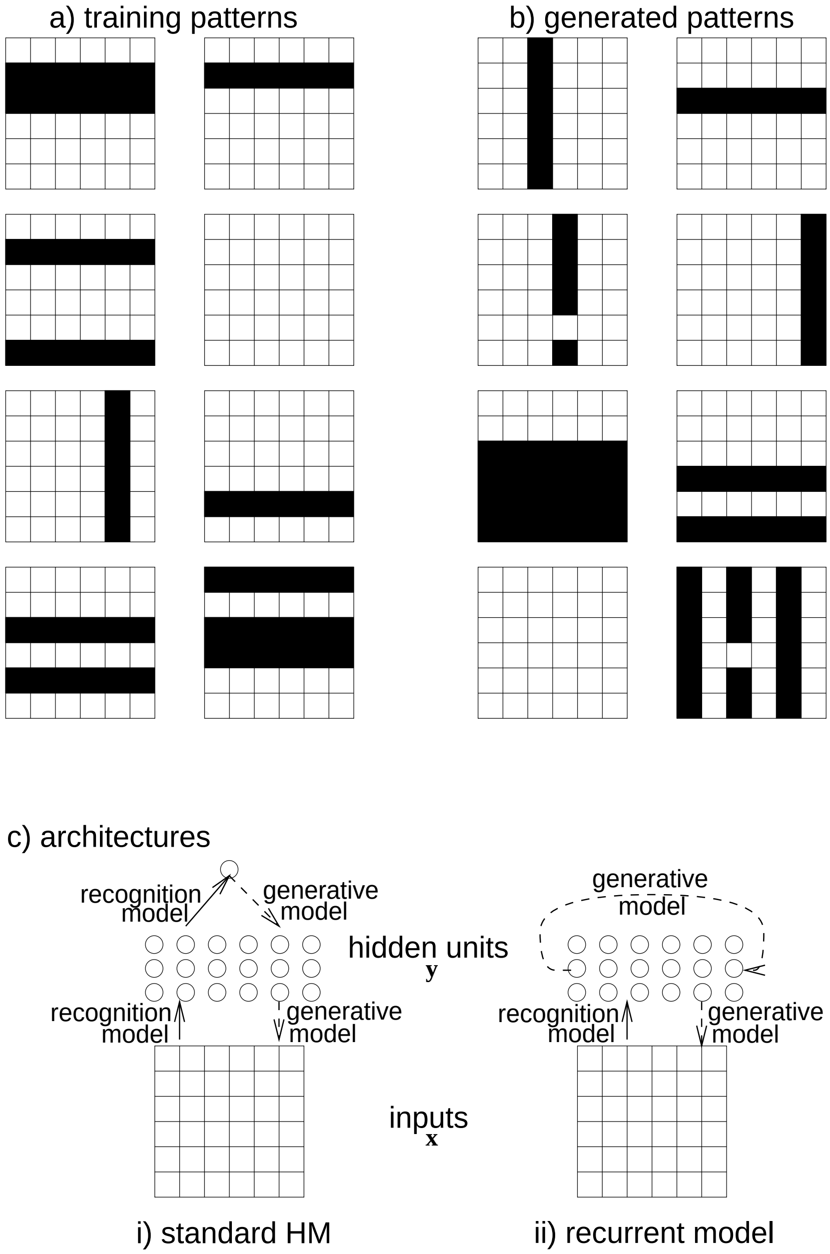

Figure 4: 6×6 binary bars patterns and network model. a) 8 random samples from the training set. b) 8 random samples drawn from the Direct method network's generative model after $100,000$ trials. The gaps in the bars show that the model is not yet quite perfect. c) Two different architectures: i) a standard hierarchical form; ii) the recurrent form, for use either with the Direct sampling method or the BM.

high learning rate. Rather than forcing positivity, we adopted the statistically-motivated competitive activation function of Dayan & Zemel (1995; see also Saund, 1995) which embodies an effective constraint that the activity of each input unit is caused on each occasion by at most one of the causes that are present, and uses weights which act like probability odds and are therefore bound to be positive.

Simulations suggest that the main effect of using a high learning rate is to encourage the network to store in the generative weights of units complete input patterns, which the wake-sleep algorithm then manipulates. However, this is an imperfect method of achieving such a result, since it only stores such patterns properly at the start of learning. Rather than do this, at random (on average, once every 5000 pattern presentations), we initialised an unused hidden unit with a pattern that the network fails to explain competently. Hidden units were considered unused if sum of their generative weights were less than $1/10$ of the maximum value across units. A pattern was deemed incompetently represented if the cost of coding the output units was more than 4 standard deviations away from the mean across recent other patterns. The algorithm is insensitive to manipulations in these parameters, although using a longer periodicity slows down learning. It is easy to see that there is a (non-zero) value of the generative bias for the added unit such that adding the unit is bound to increase the likelihood. However, it is generally impossible to know how to set this critical value. Therefore we set the generative bias arbitrarily to 1.0 and let wake-sleep modify it.[3] These modifications made wake-sleep work consistently on bars problems from 4×4 to the largest we tried, 20×20, and with either the BM or the Direct method. Figure 4b shows some samples generated by a Direct method version of the network – note that it has captured the regularity that horizontal and vertical bars do not coexist.

Finally, it would normally be substantially more work per iteration to learn BM than to learn the Direct method, because of the negative phase of BM learning (note that both phases will happen during the wake phase of the Helmholtz machine, since the recurrent model is in the generative model rather than the recognition model). However, we can take advantage of the fact that the network only has one hidden layer and perform this negative phase whilst drawing samples during sleep. This would not be possible for a hierarchical network with more than one hidden layer.

Since there are 2^{15} possible states of the hidden units, it is intractable to work out the true generative distribution for the BM (which would require calculation of the partition function) or the Direct method (would require calculation of the equilibrium distribution). This inability is, of course, orthogonal to the capacity of the network to extract the bars. Therefore, we took advantage of the fact that the recognition model does not require sampling and merely report running averages of the cost of coding just the input units. For the optimal model, this would be 0 bits, since this measure ignores the cost of coding the activities of the hidden units. Nevertheless, it is a metric of sorts for how having a

[3] Note that this means that the likelihood can *decrease* rather than increase on the introduction of the unit.

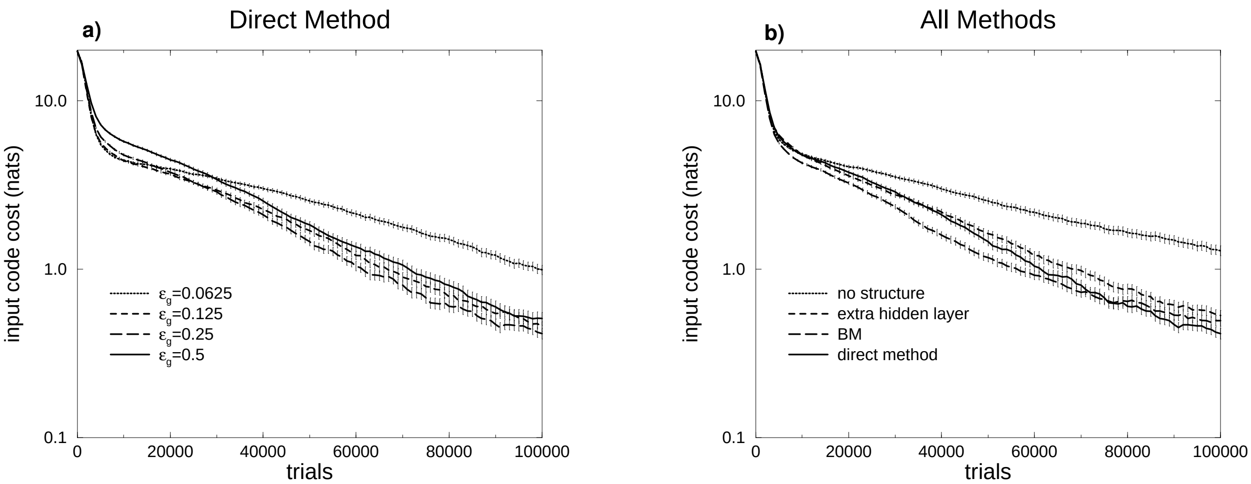

Figure 5: Learning curves for the bars problem. Both graphs show on a linear-log scale, average, low-pass filtered costs (in nats) of coding input patterns as a function of the number of training trials (together with standard errors about the mean). Averages are over 300 trials. a) Four different learning rates (ϵ_g) for the lateral connections using the Direct method. Note the relative insensitivity to the learning rate. b) Comparison of learning curves for four different methods. The architecture labelled no 'structure' does not have the representational power to capture the fact that there are *either* horizontal *or* vertical bars. Although this incapacity need not affect the cost of coding the *input* units, it evidently makes learning significantly slower.

more faithful generative model in the $\mathbf{y}$ layer helps learning of the generative model from $\mathbf{y}$ to $\mathbf{x}$.

Figure 5a shows this measure of the performance of the network for various learning rates for the lateral weights for the Direct method. Figure 5b summarises learning curves for Direct method and the BM, together with those for the standard architecture for this task (an extra hidden layer and no connections between units in the $\mathbf{y}$ layer), and an incomplete architecture without the lateral connections *or* the extra layer. We see that both direct and BM methods work quite well, and, that there is a definite advantage in having these weights even for the task of learning the mapping from $\mathbf{y}$ to $\mathbf{x}$. They perform at least as well as the fully hierarchical version of the machine.

6 Discussion

In this paper we have discussed the issue of using lateral interconnections between units to express dependencies in their activities. A Markov random field, in the form of either a Gaussian or a binary Boltzmann machine is the obvious candidate, and indeed we presented two such methods. We also suggested an alternative sampling model, which takes advantage of the key property of wake-sleep learning that, during the sleep phase, the states of all the hidden units in the network are known. This allows the use of the simple and local delta rule to learn the conditional distribution of each unit, given the states of all its peers. The delta rule is exactly the learning rule that is used in the rest of the wake-sleep algorithm, and its use here obviates the need for anything like the negative phase of the Boltzmann machine. Of course, having to perform sampling at all may incur a severe cost (but see Hinton & Ghahramani (1997) for arguments against this). We also observed that it is possible to use essentially exactly the same connections for deterministic mean-field iterations and stochastic sampling.

For the case of factor analysis, we used the models to answer a question posed by earlier work (Neal & Dayan, 1997) as to how to represent arbitrary covariance matrices in a natural way, without requiring the sort of laddered architecture seen in figure 2. Here, by using the natural gradient version of Amari (1997), the direct method and the BM have similar complexities, since one can avoid the apparent requirement for the BM of having either a sample-based negative phase of learning, or of inverting the lateral connection matrix.

We also saw how to use lateral weights within a layer to mediate dependencies within the generative model, and a particular form of Gibbs sampling to mediate dependencies within the recognition model. This form of Gibbs sampling requires computing the difference between the activities of units in a layer and the top-down prediction of those activities based on the states of units in the layer above.

The Gaussian factor analysis model is clearly a poor model for cortical representations, for instance lacking non-linearities and requiring activity levels to be both positive and negative. However, it can be useful as a metaphor to think about the roles of different aspects of cortical micro- and macro-circuitry (Rao & Ballard, 1997). One important issue is exploring ways of allowing both fast bottom-up inference and slower 'interactive' inference that integrates bottom-up and top-down information (Dayan, 1997). Thorpe *et al's* (1996) results showing that fairly complex visual recognition tasks can be accomplished in as little as $150ms$ suggests that there will not always be enough time to do extensive Gibbs sampling to explore a recognition distribution. Indeed, this is one of the advantages of the conventional Helmholtz machine with its computationally straightforward (albeit approximate) bottom-up model. However, in other cases, top-down influences are key (see Ullman (1996) for discussion).

In the context of the integrated Gaussian model in equation 16, one attractive though speculative possibility is that bottom-up connections to layer IV calculate $\mathsf{R}^{*T}\mathbf{x}$ directly, to give a first, and fast, estimate of $\mathbf{y}$. Then, if this estimate is incorrect or inadequate, or, maybe, just that there is enough time, then some form of sampling can be performed using the two sets of lateral connections. The vertical connections (between layer IV and layers II/III) mediate local interactions between cells that account for similar structure in the input; the horizontal connections between layer II/III cells in different columns represent longer range interactions and form part of the generative model as hinted at by the results of Burkhalter (1993) on the similar times of development of the lateral and top-down weights in human V1. Switching between bottom-up and integrative modes somewhat parallels the role Carpenter & Grossberg (see 1991) suggest for neuromodulators in altering dynamics in the 'hidden' layer of their adaptive resonant pattern recognisers, except that rather than eliminating a $\mathbf{y}$ unit from competition, it would allow a correct balance to be made between all possible influences on the representation $\mathbf{y}$.

We also developed sampling methods for a stochastic binary model. In this case, there is no easy shortcut for the BM, since there is no getting around the negative phase of learning. The direct method will still work (in fact this case is theoretically preferable for the direct method, since there is no possibility of divergence), and can still perfectly recover certain distributions, including ones created by a BM. A laddered architecture can do very well too, but at the cost of asymmetry. It is not possible to specify such a simple architecture to perform correct Gibbs sampling in a general binary model, since one cannot correctly account for explaining away by subtracting the predicted state of an input from the actual binary state of that input.

Acknowledgements

This work is supported by NIH grant 1 R29 MH 55541-01. I am very grateful for comments to John Hertz, Thomas Hofmann, Quaid Morris and Haim Sompolinsky. Opinions expressed are mine alone.

References

[1] Amari, S (1997). Natural gradient works efficiently in learning. Submitted to *Neural Computation.*

[2] Burkhalter, A (1993). Development of forward and feedback connections between areas V1 and V2 of human visual cortex. *Cerebral Cortex,* **3**, 476-87.

[3] Carpenter, GA & Grossberg, S (1991). *Pattern Recognition by Self-Organizing Neural Networks.* Cambridge, MA: MIT Press.

[4] Dayan, P (1997). Recognition in hierarchical models. In F Cucker & M Shub, editors, *Foundations of Computational Mathematics.* Berlin, Germany: Springer.

[5] Dayan, P, Hinton, GE, Neal, RM & Zemel, RS (1995). The Helmholtz machine. *Neural Computation,* **7**, 889-904.

[6] Dayan, P & Zemel, RS (1995). Competition and multiple cause models. *Neural Computation,* **7**, 565-579.

[7] Everitt, BS (1984). *An Introduction to Latent Variable Models.* London: Chapman and Hall.

[8] Fitzpatrick, D (1996). The functional organization of local circuits in visual cortex: insights from the study of tree shrew striate cortex. *Cerebral Cortex,* **6**, 329-341.

[9] Frey, BJ, Hinton, GE & Dayan, P (1995). Does the wake-sleep algorithm produce good density estimators? *Advances in Neural Information Processing Systems, 8,* forthcoming.

[10] Frey, BJ (1997). *Bayesian Networks for Pattern Classification, Data Compression, and Channel Coding.* PhD Thesis, Department of Electrical and Computer Engineering, University of Toronto, Toronto, Canada.

[11] Geman, S & Geman, D (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* **6**, 721-741.

[12] Ghahramani, Z & Hinton, GE (1998). Hierarchical non-linear factor analysis and topographic maps. In *Advances in Neural Information Processing, 10,* MIT Press.

[13] Gilbert, CD (1993). Circuitry, architecture, and functional dynamics of visual cortex. *Cerebral Cortex,* **3**, 373-386.

[14] Hasselmo, ME (1995). Neuromodulation and cortical function: modeling the physiological basis of behavior. *Behavioural Brain Research,* **67**, 1-27.

[15] Hasselmo, ME & Cekic, M (1996). Suppression of synaptic transmission may allow combination of associative feedback and self-organizing feedforward connections in the neocortex. *Behavioural Brain Research,* **79**, 153-161.

[16] Hinton, GE, Dayan, P, Frey, BJ & Neal, RM (1995). The wake-sleep algorithm for unsupervised neural networks. *Science,* **268**, 1158-1160.

[17] Hinton, GE & Ghahramani, Z (1997). Generative models for discovering sparse distributed representations. *Philosophical Transactions of the Royal Society, B,* to appear.

[18] Hinton, GE & Sejnowski, TJ (1986). Learning and relearning in Boltzmann machines. In DE Rumelhart, JL McClelland and the PDP research group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations,* Cambridge, MA: MIT Press, 282-317.

[19] Hinton, GE & Zemel, RS (1994). Autoencoders, minimum description length and Helmholtz free energy. In JD Cowan, G Tesauro and J Alspector, editors, *Advances in Neural Information Processing Systems 6.* San Mateo, CA: Morgan Kaufmann, 3-10.

[20] Jaakkola, T. *Variational Methods for Inference and Estimation in Graphical Models.* PhD Thesis, Department of Brain and Cognitive Sciences, MIT.

[21] Jaakkola, T, Saul, LK & Jordan, MI (1995). Fast learning by bounding likelihoods in sigmoid belief nets. *Advances in Neural Information Processing Systems, 8.*

[22] Levitt, JB, Lund, JS & Yoshioka, T (1996). Anatomical substrates for early stages in cortical processing of visual information in the macaque monkey. *Behavioural Brain Research,* **76**, 5-19.

[23] Marroquin, JL & Ramirez, A (1991). Stochastic cellular automata with Gibbsian invariant measures. *IEEE Transactions on Information Theory,* **37**, 541-551.

[24] Mumford, D (1994). Neuronal architectures for pattern-theoretic problems. In C Koch and J Davis, editors, *Large-Scale Theories of the Cortex.* Cambridge, MA: MIT Press, 125-152.

[25] Neal, RM & Dayan, P (1997). Factor Analysis using delta-rule wake-sleep learning. *Neural Computation,* in press.

[26] Olshausen, BA & Field, DJ (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature,* **381**, 607-609.

[27] Pearl, J (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* San Mateo, CA: Morgan Kaufmann.

[28] Poggio, T, Gamble, EB & Little, JJ (1988). Parallel integration of visual modules. *Science,* **242**, 436-440.

[29] Rao, PNR & Ballard, DH (1997). Dynamic model of visual memory predicts neural response properties in the visual cortex. *Neural Computation,* to appear.

[30] Rubin, DB & Thayer, DT (1982). EM algorithms for ML factor analysis. *Psychometrika,* **47**, 69-76.

[31] Saul, LK, Jaakkola, T & Jordan, MI (1995). *Mean Field Theory for Sigmoid Belief Networks.* Computational Cognitive Science Technical Report 9501, MIT.

[32] Saul, LK & Jordan, MI (1997). *A Mean Field Algorithm for Unsupervised Neural Networks.* In preparation.

[33] Saund, E (1995). A multiple cause mixture model for unsupervised learning. *Neural Computation*, **7**, 51-71.

[34] Thorpe, S, Fize, D & Marlot, C (1996). Speed of processing in the human visual system. *Nature*, **381**, 520-522.

[35] Ullman, S (1996). *High-Level Vision: Object Recognition and Visual Cognition.* Cambridge, MA: MIT Press.

[36] Zemel, RS (1994). *A Minimum Description Length Framework for Unsupervised Learning.* PhD Dissertation, Computer Science, University of Toronto, Canada.

[37] Zemel, RS & Hinton, GE (1995). Learning population codes by minimizing description length. *Neural Computation*, **7**, 549-564.