Submitted to NIP98, Algorithms and Architectures, oral presentation.

Markovian Inference in Belief Networks

Brendan J. Frey
bfrey@uiuc.edu
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
405 N. Mathews Ave., Urbana, Illinois 61801

Neil Lawrence
Neil.Lawrence@cl.cam.ac.uk
Computer Laboratory
University of Cambridge
New Museums Site, Pembroke Street
Cambridge UK CB2 3QG

Christopher M. Bishop
cmbishop@microsoft.com
Microsoft Research – Cambridge
St. George House, 1 Guildhall St.
Cambridge, UK CB2 3NH

Abstract

Bayesian belief networks can represent the complicated probabilistic processes that form natural sensory inputs. Once the parameters of the network have been learned, nonlinear inferences about the input can be made by computing the posterior distribution over the hidden units (*e.g.*, depth in stereo vision) given the input. Computing the posterior distribution exactly is not practical in richly-connected networks, but it turns out that by using a variational (a.k.a., mean field) method, it is easy to find a product-form distribution that approximates the true posterior distribution. This approximation assumes that the hidden variables are independent given the current input. In this paper, we explore a more powerful variational technique that models the posterior distribution using a Markov chain. We compare this method with inference using mean fields and mixtures of mean fields in randomly generated networks.

1 Introduction

Belief networks express how the joint distribution over a set of variables $\mathbf{s} = (s_1, \ldots, s_N)$ factors into a product of conditional distributions. This decomposition can be used to simplify inference algorithms and learning algorithms. Figure 1a shows the directed graph for a simple belief network, where each node represents a variable in $\mathbf{s}$. Taking A_i as the set of indices for the parents of s_i (those variables having edges directed to s_i), the joint distribution can be written

$$P(\mathbf{s}) \equiv \prod_{i=1}^{N} P(s_i|\{s_j\}_{j\in A_i}). \tag{1}$$

In the case of sigmoid belief networks (Neal, 1992), the variables are binary and the parameterized conditional probability that unit s_i has the value 1 is given by the logistic sigmoid function of the net input:

$$P(s_i = 1|\{s_j\}_{j\in A_i}, \boldsymbol{\theta}) \equiv 1/(1 + e^{-n_i}), \quad n_i \equiv \textstyle\sum_{j=0}^{N} \theta_{ij} s_j, \tag{2}$$

where we have introduced an extra unit s_0 that is clamped to 1 to account for a bias in the net input. The graph structure is represented by clamping some weights to 0: $\theta_{ij} \equiv 0$ if $j \notin A_i$ (except for $j = 0$). Notice that the net input n_i is a random variable given by a weighted sum of the stochastically chosen binary activities of the parents of s_i, in contrast to the mean activities used by nonlinear multilayer perceptrons. The joint distribution for a sigmoid belief network is

$$P(\mathbf{s}|\boldsymbol{\theta}) = \textstyle\prod_{i=1}^{N} e^{s_i n_i}/(1 + e^{n_i}), \quad n_i \equiv \sum_{j=0}^{N} \theta_{ij} s_j. \tag{3}$$

For a given input pattern, probabilistic inference consists of computing the posterior distribution over the unobserved hidden units $\mathbf{h}$ given the observed data $\mathbf{d}$. For example, the observed units in Figure 1a are shown in black, whereas the unobserved ones are shown in white. In general, different units may be observed in different cases, although in some learning problems such as vision, V is fixed. In richly-connected networks such as layered networks, the time needed to compute the posterior distribution exactly grows exponentially with the number of hidden units. So, we usually approximate inference using Markov chain Monte Carlo methods (Neal, 1992), recognition networks (Hinton *et al.*, 1995), loopy probability propagation (Frey, 1998), or variational techniques (Jordan *et al.*, 1998).

In a variational approximation we try to fit a parameterized distribution to the hidden units for the current input pattern. To simplify the math, we form a parameterized variational distribution over *all* of the variables, and then fix some of the parameters so that the visible units take on their observed values. The dependence of the variational distribution on the parameters $\boldsymbol{\phi}$ is indicated by writing $Q(\mathbf{s}|\boldsymbol{\phi})$. The relative entropy between $P(\mathbf{s}|\boldsymbol{\theta})$ and $Q(\mathbf{s}|\boldsymbol{\phi})$ is

$$\mathcal{F}(\boldsymbol{\phi}, \boldsymbol{\theta}) \equiv \langle \ln Q(\mathbf{s}|\boldsymbol{\phi})\rangle - \langle \ln P(\mathbf{s}|\boldsymbol{\theta})\rangle, \tag{4}$$

where $\langle\cdot\rangle$ is an expectation with respect to $Q(\cdot|\boldsymbol{\phi})$.

Denoting the hidden units for the current input pattern by $\mathbf{h}$, it is easy to show that

$$\mathcal{F}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \langle \ln Q(\mathbf{h}|\boldsymbol{\phi})\rangle - \langle \ln P(\mathbf{h}|\mathbf{d}, \boldsymbol{\theta})\rangle - \ln P(\mathbf{d}|\boldsymbol{\theta}). \tag{5}$$

(Recall that the variational parameters $\boldsymbol{\phi}$ fix the values of $\mathbf{d}$, so $\langle \ln Q(\mathbf{d}|\boldsymbol{\phi})\rangle = 0$.) This is just the relative entropy between the posterior and the variational distribution, minus a the log-likelihood of the data, which does not depend on $\boldsymbol{\phi}$. So, minimizing $\mathcal{F}(\boldsymbol{\phi}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\phi}$ is equivalent to fitting the variational distribution to the posterior.

In the "free energy" in (4), $\langle \ln P(\mathbf{s}|\boldsymbol{\theta})\rangle$ can be simplified using (3):

$$\langle \ln P(\mathbf{s}|\boldsymbol{\theta})\rangle = \textstyle\sum_{i=1}^{N}\sum_{j=0}^{N} \theta_{ij}\langle s_i s_j\rangle - \sum_{i=1}^{N}\langle \ln(1 + e^{n_i})\rangle. \tag{6}$$

Evaluation of the last term requires an exponential number of computations, so we follow Saul *et al.* (1996) and bound it. Applying the identity

$$\langle \ln(1 + e^{n_i})\rangle = \langle \ln e^{\xi_i n_i}(e^{-\xi_i n_i} + e^{(1-\xi_i)n_i})\rangle = \xi_i\langle n_i\rangle + \langle \ln(e^{-\xi_i n_i} + e^{(1-\xi_i)n_i})\rangle, \tag{7}$$

we then use Jensen's inequality to exchange the expectation and the logarithm:

$$\langle \ln(1 + e^{n_i})\rangle \leq \xi_i\langle n_i\rangle + \ln(\langle e^{-\xi_i n_i}\rangle + \langle e^{(1-\xi_i)n_i}\rangle). \tag{8}$$

This gives an upper bound $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ on $\mathcal{F}(\boldsymbol{\phi}, \boldsymbol{\theta})$ that is more tractable:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta}) \equiv & \langle \ln Q(\mathbf{s}|\boldsymbol{\phi}) \rangle + \textstyle\sum_{i=1}^{N} \sum_{j=0}^{N} \theta_{ij} \big(\xi_i \langle s_j \rangle - \langle s_i s_j \rangle \big) \\ & + \textstyle\sum_{i=1}^{N} \ln \big(\langle e^{-\xi_i n_i} \rangle + \langle e^{(1-\xi_i) n_i} \rangle \big). \end{aligned} \tag{9}$$

By minimizing $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\xi}$ and $\boldsymbol{\phi}$, we are minimizing a bound on the distance between the posterior distribution and the variational distribution.

An advantage of using $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ as a Lyapunov function for inference is that it is also an upper bound on the negative log-likelihood of the data, since $\langle \ln Q(\mathbf{h}|\boldsymbol{\phi}) \rangle - \langle \ln P(\mathbf{h}|\mathbf{d}, \boldsymbol{\theta}) \rangle$ in (5) is always positive:

$$\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta}) \geq \mathcal{F}(\boldsymbol{\phi}, \boldsymbol{\theta}) \geq -\ln P(\mathbf{d}|\boldsymbol{\theta}). \tag{10}$$

By setting some of the parameters in $\boldsymbol{\phi}$ to account for the visible units $\mathbf{d}$ and then minimizing $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\xi}$, $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$, we can perform generalized expectation maximization (Neal and Hinton, 1993).

2 Markovian inference

The mean field theory for sigmoid belief networks (Saul *et al.*, 1996) uses a product-form variational distribution over the hidden units. In this paper, we consider a variational distribution which is described by a Markov chain. The Markov chain model is more general than the factorized distribution of mean field theory, and provides an alternative approach to a mixture of mean field models (Jaakkola and Jordan, 1998; Bishop *et al.*, 1997). In order to simplify the math, we will consider Markov chains that decouple different layers in the network; *i.e.*, we effectively use one Markov chain per layer of hidden units. We can express the variational model graphically, and compare it with alternative approaches, as shown in Figure 1.

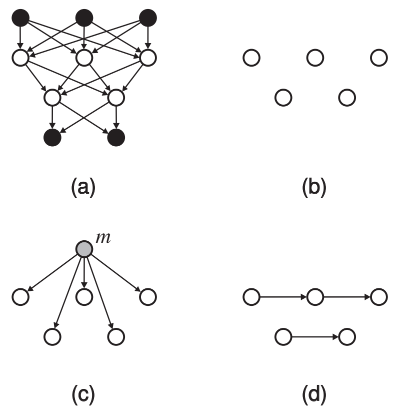

Figure 1: (a) A densely connected belief network. Clamped input and output units are black, while hidden units are white. (b) Mean field assumes a fully factorized distribution over hidden units. (c) In the mixed mean field approach, there is an additional mixture component variable shown in grey. (d) In this paper, we consider a posterior approximation corresponding to one Markov chain for each layer of hidden units.

This form of variational distribution requires that a particular ordering of the hidden units be chosen. However, in the types of networks that we are considering there is an interchange symmetry with respect to the hidden units before learning. Exchanging the labels of any two hidden units gives an equivalent network structure with identical joint distribution over the observed variables. The specific choice of ordering effectively breaks this symmetry without loss of generality. We will assume the units are ordered layer by layer.

Let the Markov chain transition matrix for s_i given the previous variable s_{i-1} be

$$\mathbf{A}_i \equiv \begin{bmatrix} Q(s_i = 0|s_{i-1} = 0) & Q(s_i = 0|s_{i-1} = 1) \\ Q(s_i = 1|s_{i-1} = 0) & Q(s_i = 1|s_{i-1} = 1) \end{bmatrix} \equiv \begin{bmatrix} 1 - a_{i0} & 1 - a_{i1} \\ a_{i0} & a_{i1} \end{bmatrix}. \tag{11}$$

We parameterize the probability a_{ij} by ϕ_{ij} using $a_{ij} \equiv 1/(1+\exp[-\phi_{ij}])$. For the sake of notational simplicity, we will assume the Markov chain weaves its way through all layers of the network, *including* the visible layers. If s_i is visible, we fix $a_{i0} = a_{i1} = 1$ if $s_i = 1$ and fix $a_{i0} = a_{i1} = 0$ if $s_i = 0$. If s_i is the first unit in a layer, we decouple it from the previous layer by constraining $a_{i0} = a_{i1}$. Finally, in order to account for the beginning of the chain ($s_0 = 1$), we define $\mathbf{A}_0 \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. The joint probability of all variables can be written

$$Q(\mathbf{s}|\boldsymbol{\phi}) = a_{11}^{s_1}(1-a_{11})^{1-s_1} \prod_{i=2}^{N} \left\{a_{i1}^{s_i}(1-a_{i1})^{1-s_i}\right\}^{s_{i-1}} \left\{a_{i0}^{s_i}(1-a_{i0})^{1-s_i}\right\}^{1-s_{i-1}} . \tag{12}$$

2.1 Evaluation of the Bound

To compute (9), we must evaluate expressions of the form $\langle s_i \rangle$, $\langle s_i s_j \rangle$, $\langle e^{-\xi_i n_i} \rangle$, and $\langle e^{(1-\xi_i)n_i} \rangle$. If s_i is observed, $\langle s_i \rangle$ is just set to the observed value. If s_i is not observed, we evaluate its marginal distribution using the standard forward algorithm:

$$\begin{aligned} Q(s_i) &= \sum_{s_1} \ldots \sum_{s_{i-1}} Q(s_i|s_{i-1}) Q(s_{i-1}|s_{i-2}) \ldots Q(s_2|s_1) Q(s_1) \\ &= \sum_{s_{i-1}} Q(s_i|s_{i-1}) \sum_{s_{i-2}} Q(s_{i-1}|s_{i-2}) \ldots \sum_{s_1} Q(s_2|s_1) Q(s_1), \end{aligned} \tag{13}$$

where we have dropped the conditioning on $\boldsymbol{\phi}$ and $\mathbf{d}$ to avoid clutter. This can be expressed as a series of matrix multiplications:

$$\begin{bmatrix} Q(s_i = 0|\boldsymbol{\phi}, \mathbf{d}) \\ Q(s_i = 1|\boldsymbol{\phi}, \mathbf{d}) \end{bmatrix} = \prod_{j=i}^{0} \mathbf{A}_j, \tag{14}$$

where $\prod_{j=i}^{0}$ indicates that matrix $\mathbf{A}_i$ appears on the far left and vector $\mathbf{A}_0$ appears on the far right. The computation of (14) takes time which is linear in the length of the chain, and hence is tractable. We then have

$$\langle s_i \rangle = [0 \;\; 1] \prod_{j=i}^{0} \mathbf{A}_j, \tag{15}$$

where the row vector $[0 \;\; 1]$ extracts the second component of the distribution.

Now consider the computation of $\langle s_i s_j \rangle$ in (9). As a consequence of the layered structure of the belief network, the prefactor θ_{ij} is nonzero only if units i and j are in different layers. Since we constrain the layers to be decoupled in the Markov chain, we need only consider i and j for which $\langle s_i s_j \rangle = \langle s_i \rangle \langle s_j \rangle$.

We are now faced with the evaluation of $\langle e^{-\xi_i n_i} \rangle$ and $\langle e^{(1-\xi_i)n_i} \rangle$ in (9). Once again, these are easily evaluated using a forward propagation along the chain:

$$\begin{aligned} &\langle e^{-\xi_i n_i} \rangle = \langle \textstyle\prod_{j=0}^{N} e^{-\xi_i \theta_{ij} s_j} \rangle \\ &= \sum_{s_1} \ldots \sum_{s_N} e^{-\xi_i \theta_{iN} s_N} Q(s_N|s_{N-1}) e^{-\xi_i \theta_{iN-1} s_{N-1}} \ldots Q(s_2|s_1) e^{-\xi_i \theta_{i1} s_1} Q(s_1) e^{-\xi_i \theta_{i0}} \\ &= \sum_{s_N} e^{-\xi_i \theta_{iN} s_N} \sum_{s_{N-1}} Q(s_N|s_{N-1}) e^{-\xi_i \theta_{iN-1} s_{N-1}} \ldots \sum_{s_1} Q(s_2|s_1) e^{-\xi_i \theta_{i1} s_1} Q(s_1) e^{-\xi_i \theta_{i0}} \\ &= [1 \;\; 1] \textstyle\prod_{j=N}^{0} \mathbf{K}_{ij} \mathbf{A}_j, \end{aligned} \tag{16}$$

and

$$\langle e^{(1-\xi_i)n_i} \rangle = [1 \;\; 1] \prod_{j=N}^{0} \widetilde{\mathbf{K}}_{ij} \mathbf{A}_j, \tag{17}$$

where we have defined

$$\mathbf{K}_{ij} \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{-\xi_i \theta_{ij}} \end{bmatrix} \qquad \widetilde{\mathbf{K}}_{ij} \equiv \begin{bmatrix} 1 & 0 \\ 0 & e^{(1-\xi_i)\theta_{ij}} \end{bmatrix}. \tag{18}$$

Finally, we need to evaluate $\langle \ln Q(\mathbf{s}|\boldsymbol{\phi})\rangle$ in (9). To do this we make use of the Markov chain property to obtain

$$\langle \ln Q(\mathbf{s}|\boldsymbol{\phi})\rangle = H(a_{11}) + \textstyle\sum_{i=2}^{N}\big\{\langle s_{i-1}\rangle H(a_{i1}) + (1 - \langle s_{i-1}\rangle)H(a_{i0})\big\}, \tag{19}$$

where we have introduced the negative binary entropy function

$$H(a_{ij}) \equiv a_{ij}\ln a_{ij} + (1 - a_{ij})\ln(1 - a_{ij}). \tag{20}$$

Using the above formulas, the time needed to compute the likelihood bound scales linearly with the number of edges in the network.

2.2 Probabilistic Inference: The Generalized E-Step

For a given input pattern $\mathbf{d}$, probabilistic inference entails setting some of the parameters in $\boldsymbol{\phi}$ to account for $\mathbf{d}$ and then minimizing $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\xi}$ and $\boldsymbol{\phi}$. To compute the derivative with respect to $\boldsymbol{\xi}$, we use the following:

$$\begin{aligned}
\frac{\partial\langle e^{-\xi_i n_i}\rangle}{\partial \xi_i} &= \sum_{k=0}^{N}[1\ \ 1]\Big(\textstyle\prod_{j=N}^{k+1}\mathbf{K}_{ij}\mathbf{A}_j\Big)\begin{bmatrix} 0 & 0 \\ 0 & -\theta_{ik}e^{-\xi_i\theta_{ik}} \end{bmatrix}\mathbf{A}_k\Big(\textstyle\prod_{j=k-1}^{0}\mathbf{K}_{ij}\mathbf{A}_j\Big) \\
\frac{\partial\langle e^{(1-\xi_i) n_i}\rangle}{\partial \xi_i} &= \sum_{k=0}^{N}[1\ \ 1]\Big(\textstyle\prod_{j=N}^{k+1}\widetilde{\mathbf{K}}_{ij}\mathbf{A}_j\Big)\begin{bmatrix} 0 & 0 \\ 0 & -\theta_{ik}e^{(1-\xi_i)\theta_{ik}} \end{bmatrix}\mathbf{A}_k\Big(\textstyle\prod_{j=k-1}^{0}\widetilde{\mathbf{K}}_{ij}\mathbf{A}_j\Big).
\end{aligned} \tag{21}$$

By storing partial products in the forward and reverse directions, all of these derivatives can be computed in time that scales linearly with the number of edges in the network.

To compute the derivative of $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ with respect to ϕ_{jk}, we need the derivatives of $\langle s_i\rangle$, $\langle e^{-\xi_i n_i}\rangle$, and $\langle e^{(1-\xi_i)n_i}\rangle$. $\partial\langle s_i\rangle/\partial\phi_{j0} = \partial\langle s_i\rangle/\partial\phi_{j1} = 0$ if s_j and s_i are in different layers. Also, from (15), $\partial\langle s_i\rangle/\partial\phi_{j0} = \partial\langle s_i\rangle/\partial\phi_{j1} = 0$ if $j > i$. Otherwise,

$$\begin{aligned}
\frac{\partial\langle s_i\rangle}{\partial\phi_{j0}} &= a_{j0}(1 - a_{j0})[0\ \ 1]\Big(\textstyle\prod_{k=i}^{j+1}\mathbf{A}_k\Big)\begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\mathbf{A}_k\Big), \\
\frac{\partial\langle s_i\rangle}{\partial\phi_{j1}} &= a_{j1}(1 - a_{j1})[0\ \ 1]\Big(\textstyle\prod_{k=i}^{j+1}\mathbf{A}_k\Big)\begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\mathbf{A}_k\Big).
\end{aligned} \tag{22}$$

The time needed to compute these derivatives for all i and j in a layer scales as the square of the number of units in the layer.

The derivatives of $\langle e^{-\xi_i n_i}\rangle$ and $\langle e^{(1-\xi_i)n_i}\rangle$ with respect to ϕ_{j0} and ϕ_{j1} are 0 if $j \neq A_i$. Otherwise,

$$\begin{aligned}
\frac{\partial\langle e^{-\xi_i n_i}\rangle}{\partial\phi_{j0}} &= a_{j0}(1 - a_{j0})[1\ \ 1]\Big(\textstyle\prod_{k=N}^{j+1}\mathbf{K}_{ik}\mathbf{A}_k\Big)\mathbf{K}_{ij}\begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\mathbf{K}_{ik}\mathbf{A}_k\Big) \\
\frac{\partial\langle e^{-\xi_i n_i}\rangle}{\partial\phi_{j1}} &= a_{j1}(1 - a_{j1})[1\ \ 1]\Big(\textstyle\prod_{k=N}^{j+1}\mathbf{K}_{ik}\mathbf{A}_k\Big)\mathbf{K}_{ij}\begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\mathbf{K}_{ik}\mathbf{A}_k\Big) \\
\frac{\partial\langle e^{(1-\xi_i) n_i}\rangle}{\partial\phi_{j0}} &= a_{j0}(1 - a_{j0})[1\ \ 1]\Big(\textstyle\prod_{k=N}^{j+1}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big)\widetilde{\mathbf{K}}_{ij}\begin{bmatrix} -1 & 0 \\ 1 & 0 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big) \\
\frac{\partial\langle e^{(1-\xi_i) n_i}\rangle}{\partial\phi_{j1}} &= a_{j1}(1 - a_{j1})[1\ \ 1]\Big(\textstyle\prod_{k=N}^{j+1}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big)\widetilde{\mathbf{K}}_{ij}\begin{bmatrix} 0 & -1 \\ 0 & 1 \end{bmatrix}\Big(\textstyle\prod_{k=j-1}^{0}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big).
\end{aligned} \tag{23}$$

The time needed to evaluate these derivatives for all units in a layer scales as the product of the number of units in the layer and the number of units in the parent layer.

Having evaluated the bound and its derivatives with respect to $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$, we can use a standard gradient-based minimization algorithm such as conjugate gradients to optimize the variational distribution.

2.3 Learning: The Generalized M-Step

Once we have optimized the variational distribution for each pattern in a training set (the generalized E-Step), we can modify the network parameters $\boldsymbol{\theta}$ (the generalized M-Step). To compute derivative of the likelihood bound $\mathcal{L}(\boldsymbol{\xi}, \boldsymbol{\phi}, \boldsymbol{\theta})$ for one input pattern, we use the following:

$$\begin{aligned}\frac{\partial\langle e^{-\xi_i n_i}\rangle}{\partial\theta_{ij}} &= [1\ \ 1]\Big(\prod_{k=N}^{j+1}\mathbf{K}_{ik}\mathbf{A}_k\Big)\begin{bmatrix}0 & 0\\ 0 & -\xi_i e^{-\xi_i\theta_{ij}}\end{bmatrix}\mathbf{A}_j\Big(\prod_{k=j-1}^{0}\mathbf{K}_{ik}\mathbf{A}_k\Big)\\ \frac{\partial\langle e^{(1-\xi_i) n_i}\rangle}{\partial\theta_{ij}} &= [1\ \ 1]\Big(\prod_{k=N}^{j+1}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big)\begin{bmatrix}0 & 0\\ 0 & (1-\xi_i)e^{(1-\xi_i)\theta_{ij}}\end{bmatrix}\mathbf{A}_j\Big(\prod_{k=j-1}^{0}\widetilde{\mathbf{K}}_{ik}\mathbf{A}_k\Big).\end{aligned} \tag{24}$$

The time needed to evaluate these derivatives for all units in a layer scales as the product of the number of units in the layer and the number of units in the parent layer.

3 Results on inference

In this section, we summarize the performance of Markovian inference in randomly drawn networks and compare it with mean field theory (Saul *et al.*, 1996) and mixtures of mean fields (Jaakkola and Jordan, 1998; Bishop *et al.*, 1997). Figure 2 shows the connectivity of the networks we used for testing inference. We used networks with 5 visible units and 5 hidden units – the number of hidden units was chosen to be small so that we could compute the true log-likelihood. Each hidden unit had a fan-out of n and was connected to nearby visible units, as shown for $n = 2$ in figure 2. For $n = 1, 2, 3, 4$ and 5, we evaluated inference performance in 100 networks constructed by drawing the parameters uniformly from the interval $[-1, 1]$ and then performing inference when all of the visible units were clamped to 0. The variational parameters for Markovian inference were initialized to give the same approximation to the posterior as given by the output of the mean field method.

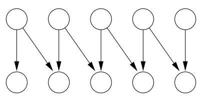

Figure 2: The connectivity of the networks used for testing inference. In this case the fan-out of the hidden units is $n = 2$.

For $n = 1$ (each hidden unit paired with a visible unit), all three inference techniques should be exact. For $n = 2$, when the visible units are clamped the hidden units form a 1st order Markov chain. For this reason, Markovian inference should be exact, whereas mean field and mixtures of mean fields should be inexact. In fact, for a fan-out of n, nth-order Markovian inference will be exact. So, for $n > 2$, the 1st order Markovian inference method described in this paper becomes inexact.

Figure 3 shows the mean percentage error in the bound (compared to the true log-likelihood) versus n, for mean field (dot-dashed), mixtures of mean fields (dotted) and Markovian inference (dashed). Markovian inference performs uniformly better for all n. We are investigating why our implementation of Markovian inference was not exact for $n = 2$ and why the mixture method was not exact for $n = 1$.

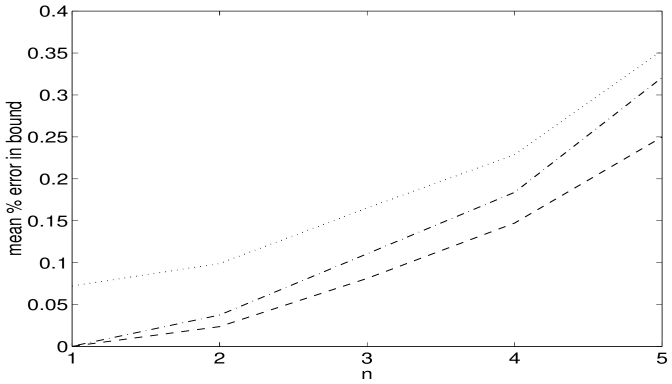

Figure 3: Mean percentage error in the bound (compared to the true log-likelihood) versus the fan-out n of the hidden units, for mean field (dot-dashed), mixtures of mean fields (dotted) and Markovian inference (dashed).

4 Conclusions

Computing the posterior distribution exactly is not practical in richly-connected belief networks, but it turns out that by using a variational (a.k.a., mean field) method, it is easy to find a product-form distribution that approximates the true posterior distribution. This approximation assumes that the hidden variables are independent given the current input. In this paper, we explored a more powerful variational technique that models the posterior distribution using a Markov chain. As expected, this "Markovian inference" method performed better at inference than the mean field method. We are currently investigating the performance of the learning algorithm.

References

Bishop, C. M., T. Jaakkola, M. I. Jordan, and N. Lawrence (1997). Approximating posterior distributions in belief networks using mixtures. Technical report, Neural Computing Research Group, Aston University, Birmingham, UK. To appear in Proceedings NIPS'97.

Frey, B. J. (1998). *Graphical Models for Machine Learning and Digital Communication*. Cambridge MA.: MIT Press. See `http://www.cs.utoronto.ca/~frey`.

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

Jaakkola, T. S. and M. I. Jordan (1998). Approximating posteriors via mixture models. In M. I. Jordan (Ed.), *Learning and Inference in Graphical Models*. Norwell MA.: Kluwer Academic Publishers.

Jordan, M. I., Z. Ghahramani, T. S. Jaakkola, and L. K. Saul (1998). An introduction to variational methods for graphical models. In M. I. Jordan (Ed.), *Learning and Inference in Graphical Models*. Norwell MA.: Kluwer Academic Publishers.

Neal, R. M. (1992). Connectionist learning of belief networks. *Artificial Intelligence* **56**, 71–113.

Neal, R. M. and G. E. Hinton (1993). A new view of the EM algorithm that justifies incremental and other variants. Unpublished manuscript available over the internet by ftp at `ftp://ftp.cs.utoronto.ca/pub/radford/em.ps.Z`.

Saul, L. K., T. Jaakkola, and M. I. Jordan (1996). Mean field theory for sigmoid belief networks. *Journal of Artificial Intelligence Research* **4**, 61–76.