Learning in Graphical Models

Hennie de Villiers

Computer Science Division, Stellenbosch University

1 These slides are adapted from those accompanying the book Bayesian Reasoning and Machine Learning. The book and demos can be downloaded from www.cs.ucl.ac.uk/staff/D.Barber/brml. We acknowledge David Barber for providing the original slides.
Introduction

- Find model parameters $\theta$ that fit the training data $d$ really well.
- Learning using maximum likelihood estimation (MLE): Maximize $p(d|\theta)$.
- But, we can perform full Bayesian estimation of the posterior over $\theta$ through inference (treating $\theta$ as if they are just normal, but unobserved variables).
- Beautiful, unified way of treating model parameters.
- Allows us to add prior information about $\theta$ easily.
- Learning with no hidden data.
- Learning with hidden data.
Learning the bias of a coin

\[ v^n = \begin{cases} 
1 & \text{if on toss } n \text{ the coin comes up heads} \\
0 & \text{if on toss } n \text{ the coin comes up tails} 
\end{cases} \]

Our aim is to estimate the probability \( \theta \) that the coin will be a head,
\[ p(v^n = 1|\theta) = \theta \] — called the ‘bias’ of the coin.

Building a model
The variables are \( v^1, \ldots, v^N \) and \( \theta \) and we require a model of the probabilistic interaction of the variables, \( p(v^1, \ldots, v^N, \theta) \). Assuming there is no dependence between the observed tosses, except through \( \theta \), we have the belief network

\[ p(v^1, \ldots, v^N, \theta) = p(\theta) \prod_{n=1}^{N} p(v^n|\theta) \]

Figure: (a): Belief network for coin tossing model. (b): Plate notation equivalent of (a). A plate replicates the quantities inside the plate a number of times as specified in the plate.
The prior

We still need to fully specify the prior $p(\theta)$. To avoid complexities resulting from continuous variables, we’ll consider a discrete $\theta$ with only three possible states, $\theta \in \{0.1, 0.5, 0.8\}$. Specifically, we assume

$$p(\theta = 0.1) = 0.15, \ p(\theta = 0.5) = 0.8, \ p(\theta = 0.8) = 0.05$$

![Graph showing discrete values of $\theta$]
The posterior

Likelihood:

\[ p(v|\theta) = \theta^v (1 - \theta)^{1 - v} \]

where \( v = 1 \) means coin came up heads, \( v = 0 \) means coin came up tails. Posterior:

\[
p(\theta|v^1, \ldots, v^N) \propto p(\theta) \prod_{n=1}^{N} p(v^n|\theta)
\]

\[
= p(\theta) \prod_{n=1}^{N} \theta^{[v^n=1]} (1 - \theta)^{[v^n=0]}
\]

\[
\propto p(\theta) \theta^{\sum_{n=1}^{N} [v^n=1]} (1 - \theta)^{\sum_{n=1}^{N} [v^n=0]}
\]

Hence

\[
p(\theta|v^1, \ldots, v^N) \propto p(\theta) \theta^{N_H} (1 - \theta)^{N_T}
\]

\[ N_H = \sum_{n=1}^{N} [v^n = 1] \] is the number of occurrences of heads.

\[ N_T = \sum_{n=1}^{N} [v^n = 0] \] is the number of tails.
For an experiment with $N_H = 2$, $N_T = 8$, the posterior distribution is

If we were asked to choose a single *a posteriori* most likely value for $\theta$, it would be $\theta = 0.5$, although our confidence in this is low since the posterior belief that $\theta = 0.1$ is also appreciable. This result is intuitive since, even though we observed more Tails than Heads, our prior belief was that it was more likely the coin is fair.
The coin posterior

Repeating the above with $N_H = 20$, $N_T = 80$, the posterior changes to

\[
\theta_0 = 0.10 \\
\theta_0 = 0.50 \\
\theta_0 = 0.80
\]

so that the posterior belief in $\theta = 0.1$ dominates. There are so many more tails than heads that this is unlikely to occur from a fair coin. Even though we \textit{a priori} thought that the coin was fair, \textit{a posteriori} we have enough evidence to change our minds.

The posterior effect

Note that in both examples, $N_T/N_H = 4$, although in the latter we are much more confident that $\theta = 0.1$.
Continuous Parameters

We first examine the case of a ‘flat’ prior $p(\theta) = k$ for some constant $k$. For continuous variables, normalisation requires

$$\int_0^1 p(\theta) d\theta = k = 1$$

Repeating the previous calculations with this flat continuous prior, we have

$$p(\theta|\mathcal{V}) = \frac{1}{c} \theta^{N_H} (1 - \theta)^{N_T}$$

where $c$ is a constant to be determined by normalisation,

$$c = \int_0^1 \theta^{N_H} (1 - \theta)^{N_T} d\theta \equiv B(N_H + 1, N_T + 1)$$

where $B(\alpha, \beta)$ is the Beta function.

Figure: Posterior $p(\theta|\mathcal{V})$ assuming a flat prior on $\theta$. (blue) $N_H = 2, N_T = 8$ and (red) $N_H = 20, N_T = 80$. The Maximum A Posteriori setting is $\theta = 0.2$ in both cases, this being the value of $\theta$ for which the posterior attains its highest value.
Beta Distribution / Function

**Definition 8.23** (Beta Distribution).

\[ p(x|\alpha, \beta) = B(x|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1} (1 - x)^{\beta-1}, \quad 0 \leq x \leq 1 \]  

(8.3.17)

where the Beta function is defined as

\[ B(\alpha, \beta) = \frac{\Gamma(\alpha) \Gamma(\beta)}{\Gamma(\alpha + \beta)} \]  

(8.3.18)

---

**Figure 8.4:** Beta distribution. The parameters \( \alpha \) and \( \beta \) can also be written in terms of the mean and variance, leading to an alternative parameterisation, see exercise(8.16).
Using a conjugate prior

For the coin tossing case, it is clear that if the prior is of the form of a Beta distribution, then the posterior will be of the same parametric form:

\[
p(\theta) = \frac{1}{B(\alpha, \beta)} \theta^{\alpha-1} (1 - \theta)^{\beta-1}
\]

the posterior is

\[
p(\theta|V) \propto \theta^{\alpha-1} (1 - \theta)^{\beta-1} \theta^{N_H} (1 - \theta)^{N_T}
\]

so that

\[
p(\theta|V) = \frac{1}{B(\alpha + N_H, \beta + N_T)} \theta^{\alpha+N_H-1} (1 - \theta)^{\beta+N_T-1}
\]

\[
\equiv B(\theta|\alpha + N_H, \beta + N_T)
\]

The prior and posterior are of the same form (both Beta distributions) but simply with different parameters. Hence the Beta distribution is ‘conjugate’ to the Binomial distribution.
Maximum Likelihood Training of Belief Networks

Consider the following model of the relationship between exposure to asbestos \( a \), being a smoker \( s \) and the incidence of lung cancer \( c \)

\[
p(a, s, c) = p(c|a, s)p(a)p(s)
\]

Each variable is binary, \( \text{dom}(a) = \{0, 1\} \), \( \text{dom}(s) = \{0, 1\} \), \( \text{dom}(c) = \{0, 1\} \). Furthermore, we assume that we have a list of patient records, where each row represents a patient’s data.

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>s</td>
<td>c</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

A database containing information about the Asbestos exposure (1 signifies exposure), being a Smoker (1 signifies the individual is a smoker), and lung Cancer (1 signifies the individual has lung Cancer). Each row contains the information for an individual, so that there are 7 individuals in the database.
To learn the table entries $p(c|a, s)$ we can do so by counting the number of times $c$ is in state $1$ for each of the 4 parental states of $a$ and $s$:

$$
\begin{align*}
    p(c = 1|a = 0, s = 0) &= 0, \\
    p(c = 1|a = 0, s = 1) &= 0.5, \\
    p(c = 1|a = 1, s = 0) &= 0.5, \\
    p(c = 1|a = 1, s = 1) &= 1
\end{align*}
$$

Similarly, based on counting, $p(a = 1) = 4/7$, and $p(s = 1) = 4/7$. These three CPTs then complete the full distribution specification.
It can be shown that for maximum likelihood training for belief nets: The table entry $p(x_i | \text{pa}(x_i))$ can be set by counting the number of times the state \( \{ x_i = s, \text{ pa}(x_i) = t \} \) occurs in the dataset (where $t$ is a vector of parental states). The table is then given by the relative number of counts of being in state $s$ compared to the other states $s'$, for fixed joint parental state $t$. 
Naive Bayes Classifier

A joint model of observations $x$ and the corresponding class label $c$ using a Belief network of the form

$$p(x, c) = p(c) \prod_{i=1}^{D} p(x_i | c)$$

Figure: Naive Bayes classifier. (a): The central assumption is that given the class $c$, the attributes $x_i$ are independent. (b): Assuming the data is i.i.d., Maximum Likelihood learns the optimal parameters of the distribution $p(c)$ and the class-dependent attribute distributions $p(x_i | c)$.

Coupled with a suitable choice for each conditional distribution $p(x_i | c)$, we can then use Bayes’ rule to form a classifier for a novel attribute vector $x^*$:

$$p(c | x^*) = \frac{p(x^* | c) p(c)}{p(x^*)} = \frac{p(x^* | c) p(c)}{\sum_c p(x^* | c) p(c)}$$
Naive Bayes example

Consider the following vector of attributes:

(likes shortbread, likes lager, drinks whiskey, eats porridge, watched England play football)

Together with each vector $x$, there is a label $nat$ describing the nationality of the person, $\text{dom}(nat) = \{\text{scottish}, \text{english}\}$.

We can use Bayes’ rule to calculate the probability that $x$ is Scottish or English:

$$
p(\text{scottish}|x) = \frac{p(x|\text{scottish})p(\text{scottish})}{p(x)}
$$

$$
= \frac{p(x|\text{scottish})p(\text{scottish})}{p(x|\text{scottish})p(\text{scottish}) + p(x|\text{english})p(\text{english})}
$$

For $p(x|nat)$ under the Naive Bayes assumption:

$$
p(x|nat) = p(x_1|nat)p(x_2|nat)p(x_3|nat)p(x_4|nat)p(x_5|nat)
$$
Using Maximum Likelihood we have: $p(\text{scottish}) = \frac{7}{13}$ and $p(\text{english}) = \frac{6}{13}$.

\[
p(x_1 = 1|\text{english}) = \frac{1}{2} \quad p(x_1 = 1|\text{scottish}) = 1 \\
p(x_2 = 1|\text{english}) = \frac{1}{2} \quad p(x_2 = 1|\text{scottish}) = \frac{4}{7} \\
p(x_3 = 1|\text{english}) = \frac{1}{3} \quad p(x_3 = 1|\text{scottish}) = \frac{3}{7} \\
p(x_4 = 1|\text{english}) = \frac{1}{2} \quad p(x_4 = 1|\text{scottish}) = \frac{5}{7} \\
p(x_5 = 1|\text{english}) = \frac{1}{2} \quad p(x_5 = 1|\text{scottish}) = \frac{3}{7}
\]

For $x = (1, 0, 1, 1, 0)^T$, we get

\[
p(\text{scottish}|x) = \frac{1 \times \frac{3}{7} \times \frac{3}{7} \times \frac{5}{7} \times \frac{4}{7} \times \frac{7}{13}}{1 \times \frac{3}{7} \times \frac{3}{7} \times \frac{5}{7} \times \frac{4}{7} \times \frac{7}{13} + \frac{1}{2} \times \frac{1}{2} \times \frac{1}{3} \times \frac{1}{2} \times \frac{1}{2} \times \frac{6}{13}} = 0.8076
\]

Since this is greater than 0.5, we would classify this person as being Scottish.
Bayesian Belief Net training

We continue with the Asbestos, Smoking, Cancer scenario,

\[ p(a, c, s) = p(c|a, s)p(a)p(s) \]

and a set of visible observations, \( V = \{(a^n, s^n, c^n), n = 1, \ldots, N\} \). With all variables binary we have parameters such as

\[ p(a = 1|\theta_a) = \theta_a, \quad p(c = 1|a = 0, s = 1, \theta_c) = \theta_c^{0,1} \]

The parameters are

\[ \theta_a, \theta_s, \theta_c^{0,0}, \theta_c^{0,1}, \theta_c^{1,0}, \theta_c^{1,1} \]

In Bayesian learning of BNs, we need to specify a prior on the joint table entries. Since in general dealing with multi-dimensional continuous distributions is computationally problematic, it is useful to specify only uni-variate distributions in the prior. As we show below, this has a pleasing consequence that for i.i.d. data the posterior also factorises into uni-variate distributions.
Global parameter independence

A convenient assumption is that the prior factorises over parameters. For our Asbestos, Smoking, Cancer example, we assume

$$p(\theta_a, \theta_s, \theta_c) = p(\theta_a)p(\theta_s)p(\theta_c)$$

Assuming the data is i.i.d., we then have the joint model

$$p(\theta_a, \theta_s, \theta_c, V) = p(\theta_a)p(\theta_s)p(\theta_c) \prod_n p(a^n|\theta_a)p(s^n|\theta_s)p(c^n|s^n, a^n, \theta_c)$$

Learning then corresponds to inference of

$$p(\theta_a, \theta_s, \theta_c|V) = \frac{p(V|\theta_a, \theta_s, \theta_c)p(\theta_a, \theta_s, \theta_c)}{p(V)} = \frac{p(V|\theta_a, \theta_s, \theta_c)p(\theta_a)p(\theta_s)p(\theta_c)}{p(V)}$$

The posterior also factorises, since

$$p(\theta_a, \theta_s, \theta_c|V) \propto p(\theta_a, \theta_s, \theta_c, V)$$

$$= \left\{ p(\theta_a) \prod_n p(a^n|\theta_a) \right\} \left\{ p(\theta_s) \prod_n p(s^n|\theta_s) \right\} \left\{ p(\theta_c) \prod_n p(c^n|s^n, a^n, \theta_c) \right\}$$

$$\propto p(\theta_a|V_a)p(\theta_s|V_s)p(\theta_c|V_c)$$
If we further assume that the prior for the table factorises over all states $a, c$:

$$p(\theta_c) = p(\theta_c^{0,0})p(\theta_c^{1,0})p(\theta_c^{0,1})p(\theta_c^{1,1})$$

then the posterior

$$p(\theta_c | V_c) \propto p(V_c | \theta_c)p(\theta_c^{0,0})p(\theta_c^{1,0})p(\theta_c^{0,1})p(\theta_c^{1,1})$$

$$= \left[ \theta_c^{0,0} \right]^\#(a=0,s=0)p(\theta_c^{0,0})\left[ \theta_c^{0,1} \right]^\#(a=0,s=1)p(\theta_c^{0,1})\left[ \theta_c^{1,0} \right]^\#(a=1,s=0)p(\theta_c^{1,0})\left[ \theta_c^{1,1} \right]^\#(a=1,s=1)p(\theta_c^{1,1})$$

$$\propto p(\theta_c^{0,0} | V_c)p(\theta_c^{0,1} | V_c)$$

$$\times \left[ \theta_c^{1,0} \right]^\#(a=1,s=0)p(\theta_c^{1,0})\left[ \theta_c^{1,1} \right]^\#(a=1,s=1)p(\theta_c^{1,1})$$

$$\propto p(\theta_c^{1,0} | V_c)p(\theta_c^{1,1} | V_c)$$

so that the posterior also factorises over the parental states of the local conditional table.
Global and Local independence

The important thing here: We can now do normal inference to get the table parameters, isn’t that nice :)
Using a Beta prior

JUST KNOW YOU CAN DO THIS

\[ p(\theta_a) = B(\theta_a | \alpha_a, \beta_a) = \frac{1}{B(\alpha_a, \beta_a)} \theta_a^{\alpha_a - 1} (1 - \theta_a)^{\beta_a - 1} \]

for which the posterior is also a Beta distribution:

\[ p(\theta_a | V_a) = B(\theta_a | \alpha_a + \#(a = 1), \beta_a + \#(a = 0)) \]

The marginal table is given by

\[ p(a = 1 | V_a) = \int_{\theta_a} p(\theta_a | V_a) \theta_a = \frac{\alpha_a + \#(a = 1)}{\alpha_a + \#(a = 1) + \beta_a + \#(a = 0)} \]

hyperparameters

The prior parameters \( \alpha_a, \beta_a \) are called hyperparameters. If one had no preference, one would set \( \alpha_a = \beta_b = 1 \).
Bayes vs Maximum Likelihood

\[ p(a = 1 | \mathcal{V}_a) = \int_{\theta_a} p(\theta_a | \mathcal{V}_a) \theta_a = \frac{\alpha_a + \#(a = 1)}{\alpha_a + \#(a = 1) + \beta_a + \#(a = 0)} \]

Corresponds in this case to adding ‘pseudo counts’ to the data.

---

**No data limit**

The posterior is just the prior. The marginal probability table corresponds to the prior ratios:

\[ p(a = 1) = \frac{\alpha_a}{\alpha_a + \beta_a} \]

For a flat prior \( \alpha = \beta = 1 \), \( p(a = 1) = 0.5 \).

---

**Infinite data limit**

The posterior is just the likelihood. The marginal probability tables are dominated by the data counts:

\[ p(a = 1 | \mathcal{V}) \rightarrow \frac{\#(a = 1)}{\#(a = 1) + \#(a = 0)} \]

which corresponds to the Maximum Likelihood solution.
Summary

- Maximum Likelihood in general corresponds to the intuitive use of ‘counting’ to set tables.
- When there are no counts of a particular configuration, the learned probabilities are zero. This can have severe effects in classifiers such as Naive Bayes.
- The Bayesian approach places priors on the tables.
- Convenient to assume global parameter independence since then the posterior factorises over the tables (assuming i.i.d.).
- Convenient also to assume local parameter independence of each conditional since then the posterior table factorises over its parental states.
- A very simple classifier is Naive Bayes. A Bayesian treatment is equivalent to using ‘pseudo counts’ and avoids overfitting.
- Naive Bayes is extremely popular. (Spam filtering, credit scoring, ....)
Hidden Variables and Missing Data

---

**Missing Data**
In practice data entries are often missing resulting in incomplete information to specify a likelihood.

---

**Observational Variables**
Observational variables may be split into visible (those for which we actually know the state) and missing (those whose states would nominally be known but are missing for a particular datapoint).

---

**Latent Variables**
Another scenario in which not all variables in the model are observed are the so-called hidden or latent variable. In this case there are variables which are essential for the model description but never observed. For example, the underlying physics of a model may contain latent processes which are essential to describe the model, but cannot be directly measured.
In learning the parameters of models we previously assumed we have complete information to define all variables of the joint model of the data \( p(v|\theta) \).

---

**Complete data**

Consider the Asbestos-Smoking-Cancer network. In the case of complete data, the likelihood is

\[
p(v^n|\theta) = p(a^n, s^n, c^n|\theta) = p(c^n|a^n, s^n, c|\theta) p(a^n|\theta_a) p(s^n|\theta_s)
\]

which is factorised in terms of the table entry parameters. We exploited this property to show that table entries \( \theta \) can be learned by considering only local information, both in the Maximum Likelihood and Bayesian frameworks.
Missing data

Now consider the case that for some of the patients, only partial information is available. For example, for patient $n$ with record $v^n = \{c = 1, s = 1\}$ it is known that the patient has cancer and is a smoker, but whether or not they had exposure to asbestos is unknown. Since we can only use the ‘visible’ available information it would seem reasonable to assess parameters using the marginal likelihood

$$p(v^n | \theta) = \sum_a p(a, s^n, c^n | \theta) = \sum_a p(c^n | a, s^n, \theta_c)p(a | \theta_a)p(s^n | \theta_s)$$

The likelihood cannot be written as a product of functions, one for each separate parameter. In this case the maximisation of the likelihood is more complex since the parameters of different tables are coupled.

Bayesian learning

A similar complication holds for Bayesian learning. Under a prior factorised over each CPT $\theta$, the posterior is also factorised. However, in the case of unknown asbestos exposure, a term $p(v^n | \theta)$ as above is introduced, which cannot be written as a product of a functions of $f_s(\theta_s)f_a(\theta_a)f_c(\theta_c)$. The missing variable therefore introduces dependencies in the posterior parameter distribution, making the posterior more complex.
Maximum Likelihood

- For hidden variables $h$, and visible variable $v$ we still have a well defined likelihood

\[ p(v|\theta) = \sum_h p(v, h|\theta) \]

- Our task is to find the parameters $\theta$ that optimise $p(v|\theta)$.
- This task is more numerically complex than in the case when all the variables are visible.
- Nevertheless, we can perform numerical optimisation using any routine we wish to find $\theta$.
- The Expectation-Maximisation algorithm is an alternative optimisation algorithm that can be very useful in producing simple and elegant updates for $\theta$ that converge to a local optimum.
- Just to hammer this home: We don’t ‘need’ the EM algorithm, but it can be very handy.
Expectation Maximization (EM)

- Initial random choice for $\theta$.
- Find expect value of hidden variables given this $\theta$.
- Maximize (for $\theta$) the likelihood of the visible data, and the hidden variables (clamped to the expected values), given the parameter $\theta$.
- Repeat

**Figure:** (a): The log likelihood. (b):Contours of the lower bound $LB(q(h = 2), \theta)$. For an initial choice $q(h = 2) = 0.5$ and $\theta = 1.9$, successive updates of the E (vertical) and M (horizontal) steps are plotted. (c): Starting at $\theta = 1.95$, the EM algorithm converges to a local optimum.
The need for optimisation

Machine learning often requires fitting a model to data. Often this means finding the parameters $\theta$ of the model that ‘best’ fit the data.

**Regression**
For example, for regression based on training data $(x^n, y^n)$ we might have a model $y(x|\theta)$ and wish to set $\theta$ by minimising

$$E(\theta) = \sum_n (y^n - y(x^n|\theta))^2$$

**Complexity**
In all but very simple cases, it is extremely difficult to find an algorithm that will guarantee to find the optimal $\theta$. 
Gradient Descent

We wish to find $x$ that minimises $f(x)$. For general $f$ there is no closed-form solution to this problem and we typically resort to iterative methods. For $x_{k+1} \approx x_k$,

$$f(x_{k+1}) \approx f(x_k) + (x_{k+1} - x_k)^T \nabla f(x_k)$$

setting

$$x_{k+1} - x_k = -\epsilon \nabla f(x_k)$$

gives

$$f(x_{k+1}) \approx f(x_k) - \epsilon |\nabla f(x_k)|^2$$

Hence, for a small $\epsilon$, the algorithm

$$x_{k+1} = x_k - \epsilon \nabla f(x_k)$$

decreases $f$. We iterate until convergence.
Gradient Descent in Action
Learning with Hidden Data: Markov Networks

If joint has the form:

\[ p(v, h|\theta) = \frac{1}{Z(\theta)} \exp(\phi(v, h|\theta)) \]

Then for independent, identically distributed data, the log likelihood is:

\[ p(v|\theta) = L(\theta) = \sum_n \left( \log \sum_h \exp \phi(v^n, h|\theta) - \log \sum_{h,v} \exp \phi(v, h|\theta) \right) \]

which has the gradient:

\[ \frac{\partial}{\partial \theta} L = \sum_n \left( \langle \frac{\partial}{\partial \theta} \phi(v^n, h|\theta) \rangle_{p(h|v^n, \theta)} - \langle \frac{\partial}{\partial \theta} \phi(v, h|\theta) \rangle_{p(h,v|\theta)} \right) \]

clamped average (awake, sees data)  free average (asleep, dreams about v)
Learning with Hidden Data: Markov Networks

\[ \frac{\partial}{\partial \theta} L = \sum_n \left( \left\langle \frac{\partial}{\partial \theta} \phi(v^n, h|\theta) \right\rangle_{p(h|v^n, \theta)} - \left\langle \frac{\partial}{\partial \theta} \phi(v, h|\theta) \right\rangle_{p(h,v|\theta)} \right) \]

- We can get the averages using sampling methods.
- Can take a loooong time for unconstrained networks (see General Boltzmann Machines with hidden units).
- Special restrictions on network make it feasible! (see Restricted Boltzmann Machines)