Probabilistic Foundations of Artificial Intelligence

Problem Set 5

Nov 14, 2014

1. Gibbs sampling

In this exercise, you will implement a Gibbs sampling algorithm for performing approximate inference in Bayesian networks. Although using a factor graph is not necessary for Gibbs sampling, we will use the already available factor graph representation from the previous exercise to conveniently acquire the Markov blanket of each variable. That way, all information required to compute the posterior distribution of a variable v given some values for all other variables, is contained in the CPTs of the neighboring factor nodes $\mathcal{N}(v)$.

More concretely, let $\boldsymbol{x}_{-v}$ be the set of all variables except for v, and $\boldsymbol{s}_{-v}$ be the value of those variables at the current iteration. Similarly, let $\boldsymbol{x}_{f\setminus v}, \boldsymbol{s}_{f\setminus v}$ be all variables that participate in factor f except for v, and $\boldsymbol{s}_{f\setminus v}$ be the values thereof. Then, to update the value of v you will have to draw from the posterior

$$P(v = d \mid \boldsymbol{x}_{-v} = \boldsymbol{s}_{-v}) = \frac{1}{Z} \prod_{f \in \mathcal{N}(v)} f(v = d, \boldsymbol{x}_{f\setminus v} = \boldsymbol{s}_{f\setminus v}),$$

where Z is a normalization factor. In practice, you will compute the above product (without the $1/Z$ part) for all $d \in \mathrm{dom}(v)$, then normalize to get a proper distribution, and finally draw from that distribution to obtain a new value for v.

You are provided some skeleton Python code in the `.zip` file accompanying this document. Take the following steps for this exercise.

(i) Install the Python dependencies listed in `README.txt`, if your system does not already satisfy them. After that, you should be able to run `demo.py` and produce some plots, albeit wrong ones for now.

(ii) Implement the missing code in `sampling.py` marked with `TODO`. In particular, you have to fill in the part that computes the posterior distribution discussed above, as well as the part that picks a variable and updates the state of the Gibbs sampler.

(iii) If your implementation is correct, you should get (approximately) correct results for the naive Bayes model of the demo file that represents the coin flipping network of exercise 2 in Problem Set 2.

(iv) Now, you can try out your Gibbs sampler on the earthquake network of the previous exercise. Compare your results to those you obtained using belief propagation. There are three parameters you can tune:

- The starting state of the Gibbs sampler. By default, it is created by drawing independent and uniformly random values for each variable.

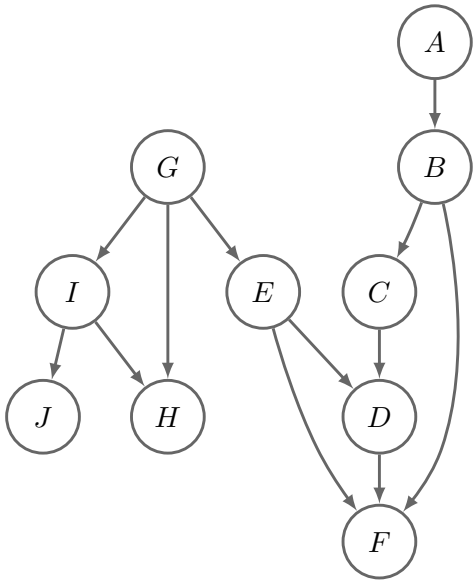


Figure 1: Bayesian network for variable elimination.

- The length of the burn-in period, during which the state is updated, but not saved. Therefore, anything sampled during this stage has no effect on the approximate marginals computed afterwards.
- The function used to obtain the approximate marginals that are plotted. By default, this function is a cumulative average, i.e., it computes the approximate marginal distribution of a variable at step i by looking at the average number of occurences of each value of that variable among the samples obtained by the algorithm up to step i. A simple modification would be to only use every k-th sample when computing these averages, since successive samples are heavily correlated.

2. Variable elimination

You are given the Bayesian network shown in Figure 1.

(i) Create a factor graph corresponding to the given network.

(ii) Suppose you want to compute $P(C \mid I = i)$ using variable elimination. Considering the variable ordering $A, B, C, D, E, F, G, H, I, J$, determine the factors that are removed and those that are created at each step of the algorithm. For example, if we define $\phi_{AB}(A, B) = P(A)P(B \mid A)$, the first step would be as shown below.

Current variable	Factors removed	Factor introduced
A	$\phi_{AB}(A, B)$	$\phi_1(A, B)$
$\vdots$	$\vdots$	$\vdots$

3. Markov chains and detailed balance

Assume that you are given a Markov chain with state space Ω and transition matrix T, which is defined for all $x, y \in \Omega$ and $t \geq 0$ as $T(x, y) := P(X_{t+1} = y \mid X_t = x)$. Furthermore, let π be the stationary distribution of the chain.

(i) Show that, if for some t the current state X_t is distributed according to the stationary distribution and additionally the chain satisfies the detailed balance equations

$$\pi(x)T(x, y) = \pi(y)T(y, x), \text{ for all } x, y \in \Omega,$$

then the following holds for all $k \geq 0$ and $x_0, \ldots, x_k \in \Omega$:

$$P(X_t = x_0, \ldots, X_{t+k} = x_k) = P(X_t = x_k, \ldots, X_{t+k} = x_0).$$

(This is why a chain that satisfies detailed balance is called *reversible*.)

(ii) Show that, if T is a symmetric matrix, then the chain satisfies detailed balance, and the uniform distribution on Ω is stationary for that chain.

4. Belief propagation on tree factor graphs*

In this exercise we will prove that the belief propagation algorithm converges to the exact marginals after a fixed number of iterations given that the factor graph is a tree, that is, given that the original Bayesian network is a polytree.

We will assume that the factor graph contains no single-variable factors. (You have already seen that if those exist, they can easily be incorporated into multi-variable factors without increasing the complexity of the algorithm.) Since the factor graph is a tree, we will designate a variable node, say a, as the root of the tree. We will consider subtrees $\mathcal{T}_{[rt]}$, where r and t are adjacent nodes in the factor graph and t is closer to the root than r, which are of two types:

- if r is a factor node (and t a variable node), then $\mathcal{T}_{[rt]}$ denotes a subtree that has t as its root, contains the whole subtree under r and, additionally, the edge $\{r, t\}$,
- if r is a variable node (and t a factor node), then $\mathcal{T}_{[rt]}$ denotes the whole subtree under r with r as its root.

See Figure 2 for two example subtrees, one of each type. The depth of a tree is defined as the maximum distance between the root and any other node. Note that, both types of subtrees $\mathcal{T}_{[rt]}$ defined above have always depths that are even numbers.

We will use the subscript notation $[rt]$ to refer to quantities constrained to the subtree $\mathcal{T}_{[rt]}$. In particular, we denote by $\mathcal{F}_{[rt]}$ the set of factors in the subtree and by $P_{[rt]}(x_v)$ the marginal distribution of v when we only consider the nodes of the subtree. More concretely, if r is a variable node, by the sum rule we get

$$P_{[rt]}(x_r) \simeq \sum_{\boldsymbol{x}_{[rt]\setminus\{r\}}} \prod_{f \in \mathcal{F}_{[rt]}} f(\boldsymbol{x}_f), \tag{1}$$

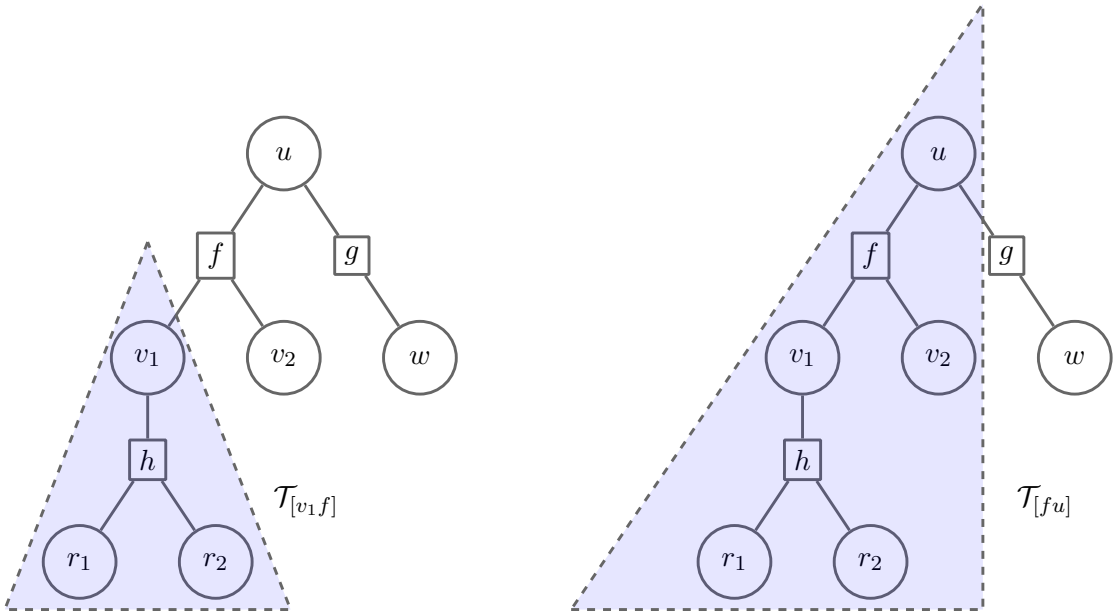


Figure 2: An example factor graph and two of its subtrees.

where $\simeq$ denotes equality up to a normalization constant.

Remember the form of the messages passed between variable and factor nodes at each iteration of the algorithm:

$$\mu_{v\to f}^{(t+1)}(x_v) := \prod_{g\in\mathcal{N}(v)\setminus\{f\}} \mu_{g\to v}^{(t)}(x_v) \tag{2}$$

$$\mu_{f\to v}^{(t+1)}(x_v) := \sum_{\boldsymbol{x}_{f\setminus\{v\}}} f(\boldsymbol{x}_f) \prod_{w\in\mathcal{N}(f)\setminus\{v\}} \mu_{w\to f}^{(t)}(x_w). \tag{3}$$

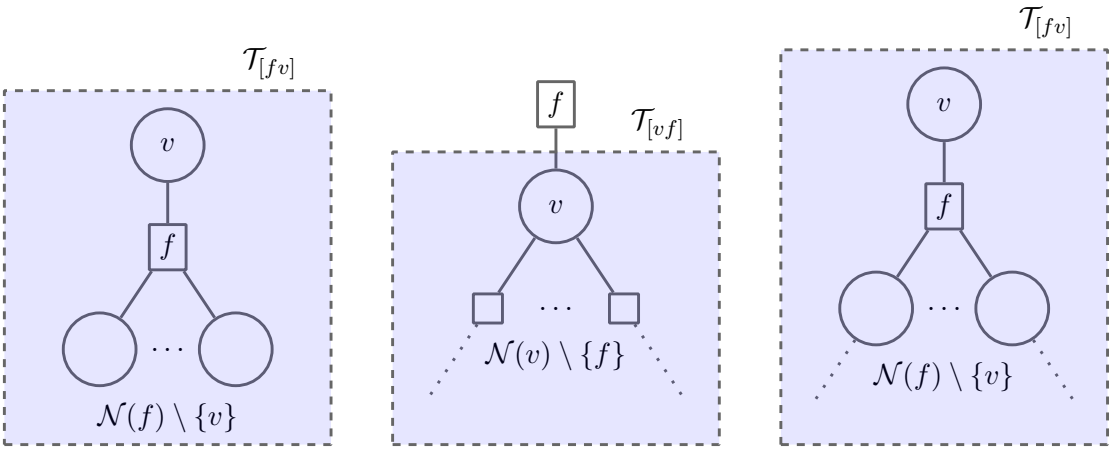
We also define the estimated marginal distribution of variable v at iteration t as

$$\hat{P}^{(t)}(x_v) := \prod_{g\in\mathcal{N}(v)} \mu_{g\to v}^{(t)}(x_v). \tag{4}$$

Our ultimate goal is to show that the estimated marginals are equal to the true marginals for all variables after a number iterations. However, we will first consider the rooted version of the factor graph and show that the previous statement holds for the root node a. More concretely, if we denote variable nodes with v and factor nodes with f, we will show using induction that for all subtrees $\mathcal{T}_{[fv]}$ of depth τ, it holds that, for all $t \geq \tau$,

$$\mu_{f\to v}^{(t)}(x_v) \simeq P_{[fv]}(x_v). \tag{5}$$

(i) Consider the base case of $\mathcal{T}_{[fv]}$ being a subtree of depth $\tau = 2$ (see Figure 3a). Show that (5) holds in this case for all $t \geq 2$.



(a) Base case. (b) First induction step. (c) Second induction step.

Figure 3: The three cases considered in the proof by induction.

(ii) Now, assume that (5) holds for all subtrees of depth $\leq \tau$. As a first step, show that for any subtree $\mathcal{T}_{[vf]}$ of depth τ (see Figure 3b) it holds that, for all $t \geq \tau + 1$,

$$\mu^{(t)}_{v \to f}(x_v) \simeq P_{[vf]}(x_v).$$

(iii) Using the result of the previous step, show that, for any subtree $\mathcal{T}_{[fv]}$ of depth $\tau' = \tau + 2$, (5) holds for all $t \geq \tau'$ (see Figure 3c).

(iv) Show that, if the factor graph rooted at a has depth d, then, for all $t \geq d$,

$$\hat{P}^{(t)}(x_a) \simeq P(x_a).$$

(v) To generalize the statement above, assume that the factor graph has diameter $\mathcal{D}$, i.e., the maximum distance between any two nodes in the graph is $\mathcal{D}$. Show that for all $t \geq \mathcal{D}$ the estimated marginal of *any* variable v is exact, that is,

$$\hat{P}^{(t)}(x_v) \simeq P(x_v).$$