Approximate Inference Methods

3007/7059 Artificial Intelligence

School of Computer Science
The University of Adelaide

Inference on Bayesian Networks

To perform inference is to make conclusions on the basis of evidence and reasoning.

We have seen how Bayesian Networks encapsulate our knowledge of the domain in terms of conditional independent assertions.

We have also studied the general principle of making statistical inferences based on Bayesian Networks.

In particular, we have studied exact inference methods on Bayesian Networks, as well as their limitation with regards to the size (in terms of the number of variables) of the network they can handle.

To make statistical inference feasible, it is essential to consider approximate inference methods.

Monte Carlo algorithms

Approximate inference methods extensively use random sampling.

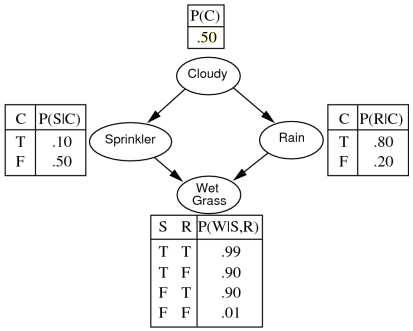
This means they belong to a large class of methods called Monte Carlo algorithms (a reference to the famous casino in Monaco), so named because the element of chance permeates these algorithms.

In general Monte Carlo approximate solutions become more accurate as the number of samples increases.

We will study two classes of approximate inference methods: Direct sampling methods and Markov Chain Monte Carlo (MCMC) methods.

Direct sampling methods

Direct sampling methods rely on the ability to generate samples from the Bayesian Network. Here is the wet grass network again:



The idea is to sample each variable in turn, in topological order (i.e. from parents to children). The probability distribution from which the value is sampled is conditioned on the parents' values (which are sampled earlier).

Direct sampling methods (cont.)

For the wet grass network, one such sampling instance could be

1. Sample from $P(Cloudy) = \langle 0.5, 0.5 \rangle$; suppose this returns $true$.
2. Sample from $P(Sprinkler|Cloudy = true) = \langle 0.1, 0.9 \rangle$; suppose this returns $false$.
3. Sample from $P(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$; suppose this returns $true$.
4. Sample from $P(WetGrass|Sprinkler = false, Rain = true) = \langle 0.9, 0.1 \rangle$; suppose this returns $true$.

This yields the sample $[true, false, true, true]$ from the network (again, the values are according to the topological order).

Direct sampling methods (cont.)

Suppose we generated N samples in this manner. Let $N_{PS}(x_1, \ldots, x_n)$ be the number of times the specific event $x_1, \ldots, x_n$ was produced among the N samples (n is the number of variables in the network).

We expect the proportion of event $x_1, \ldots, x_n$ among N samples to converge to its expected value as N approaches infinity, i.e.,

$$\lim_{N \to \infty} \frac{N_{PS}(x_1, \ldots, x_n)}{N} = P(x_1, \ldots, x_n)$$

For example, the probability of producing event $[true, false, true, true]$ is simply

$$P(true, false, true, true) = 0.5 \times 0.9 \times 0.8 \times 0.9 = 0.324$$

Hence we expect for a sufficiently large number of samples N, approximately 32.4% of them to be of this event.

Sampling partial events

We frequently want to estimate the probability of partially specified events $P(x_1, \ldots, x_m)$ with $m < n$.

This can be approximated by

$$P(x_1, \ldots, x_m) \approx \frac{N_{PS}(x_1, \ldots, x_m)}{N}$$

where $N_{PS}(x_1, \ldots, x_m)$ is now the number of samples among N whose values are consistent with $x_1, \ldots, x_m$.

Example: Estimating probability of partial events

Say the following $N = 8$ samples were generated from the wet grass network:

$$\begin{array}{c} Cloudy \\ Sprinkler \\ Rain \\ WetGrass \end{array} \begin{bmatrix} false \\ true \\ false \\ true \end{bmatrix} \begin{bmatrix} false \\ false \\ true \\ true \end{bmatrix} \begin{bmatrix} false \\ true \\ true \\ false \end{bmatrix} \begin{bmatrix} false \\ true \\ false \\ false \end{bmatrix} \begin{bmatrix} true \\ true \\ true \\ true \end{bmatrix} \begin{bmatrix} false \\ false \\ true \\ true \end{bmatrix} \begin{bmatrix} true \\ true \\ false \\ false \end{bmatrix} \begin{bmatrix} false \\ true \\ true \\ false \end{bmatrix}$$

From the samples,

$$P(Cloudy = false, Sprinkler = true, Rain = true, WetGrass = false) = \frac{2}{8} = 0.25$$

$$P(Cloudy = false, Sprinkler = true) = \frac{4}{8} = 0.5$$

$$P(Rain = true) = \frac{5}{8} = 0.625$$

Compute these probability values again via exact inference. Are there significant differences between the approximate and exact solutions? Why?

Rejection sampling

Our goal is to ultimately perform statistical inference, i.e., to compute probabilities of the form

$$P(X|e)$$

where X is the query variable, and e the observed values for the evidence variables.

Rejection sampling is a simple direct sampling method to compute an approximate value $\hat{P}(X|e)$ of the required probability.

The steps are:

1. First, N samples are generated from the network.
2. Samples which do not match the evidences e are rejected.
3. The desired estimate $\hat{P}(X = x|e)$ is obtained by counting how often $X = x$ occurs in the remaining samples.

Example: Rejection sampling

We wish to estimate

$$P(Sprinkler|Rain = true)$$

$N = 8$ samples were produced from the wet grass network

$$\begin{matrix} Cloudy \\ Sprinkler \\ Rain \\ WetGrass \end{matrix} \begin{bmatrix} false \\ true \\ false \\ true \end{bmatrix} \begin{bmatrix} false \\ false \\ true \\ true \end{bmatrix} \begin{bmatrix} false \\ true \\ true \\ false \end{bmatrix} \begin{bmatrix} false \\ true \\ false \\ false \end{bmatrix} \begin{bmatrix} true \\ true \\ true \\ true \end{bmatrix} \begin{bmatrix} false \\ false \\ true \\ true \end{bmatrix} \begin{bmatrix} true \\ true \\ false \\ false \end{bmatrix} \begin{bmatrix} false \\ true \\ true \\ false \end{bmatrix}$$

among which only 5 match the evidence $Rain = true$.

Among the remaining samples, 2 correspond to $Sprinkler = false$ and 3 correspond to $Sprinkler = true$.

Hence,

$$\hat{P}(Sprinkler|Rain = true) = \langle \frac{3}{5}, \frac{2}{5} \rangle = \langle 0.6, 0.4 \rangle$$

Compare these results to those obtained using exact inference.

Likelihood weighting

A weakness of rejection sampling is its inefficient use of sampling: It often rejects too many samples which disagree with the evidence.

This is especially true for evidences e whose prior probability $P(e)$ is very low to begin with: Only a small percentage of the samples will match the evidence.

Likelihood weighting rectifies this by generating only events that are consistent with the evidence.

Each sample is weighted by the likelihood that the event matches the evidence; This is measured by the product of the conditional probabilities for each evidence variable, given its parents.

Each sample's contribution to the desired probability is then weighted by its likelihood.

Example: Computing weights for samples

Referring to the wet grass network again, lets say we wish to obtain

$$P(Rain|Sprinkler = true, WetGrass = true)$$

Generating a sample with likelihood weighting proceeds as follows:

1. Initialise weight $w = 1$.
2. Sample from $P(Cloudy) = \langle 0.5, 0.5 \rangle$; say this returns $true$.
3. $Sprinkler$ is an evidence variable with value $true$. Thus set
   $$w \leftarrow w \times P(Sprinkler = true|Cloudy = true) = 0.1$$
4. Sample from $P(Rain|Cloudy = true) = \langle 0.8, 0.2 \rangle$; say this returns $true$.
5. $WetGrass$ is an evidence variable with value $true$. Thus set
   $$w \leftarrow w \times P(WetGrass = true|Sprinkler = true, Rain = true) = 0.099$$

This produces the sample $[true, true, true, true]$ with weight 0.099 which is tallied under $Rain = true$.

Intuitively, the weight is low because the event describes a cloudy day, which makes the sprinkler unlikely to turn on.

Computing likelihood weighted estimates

For an arbitrary Bayesian Network, let X be the query variable, e be the values of the evidence variables and Y be the unobserved variables.

Let

- $N_{WS}(X, Y, e)$ be the number of samples generated for the event X, Y and e.
- $w(X, Y, e)$ be the weight of a sample corresponding to the event X, Y and e.

The the estimate for $P(X|e)$ is

$$\hat{P}(X|e) = \alpha \sum_{\forall Y} N_{WS}(X, Y, e) w(X, Y, e)$$

Example: Computing likelihood weighted estimates

Say the following $N = 100$ samples were generated from the wetgrass network with associated likelihood/weights:

- 3 samples of $[true, true, true, true]$ with $w = 0.099$
- 2 samples of $[true, true, false, true]$ with $w = 0.09$
- 55 samples of $[false, true, true, true]$ with $w = 0.495$
- 40 samples of $[false, true, false, true]$ with $w = 0.45$

Notice that all samples are consistent with the evidence $Sprinkler = true$ and $WetGrass = true$.

The desired probability estimate is

$$\begin{aligned} & \hat{P}(Rain|Sprinkler = true, WetGrass = true) \\ = \; & \alpha\langle 3 \times 0.099 + 55 \times 0.495, 2 \times 0.09 + 40 \times 0.45\rangle \\ = \; & \alpha\langle 27.522, 18.18\rangle = \langle 0.60, 0.40\rangle \end{aligned}$$

MCMC methods

Likelihood weighting is more efficient than rejection sampling since it uses all samples generated.

However, the efficiency of likelihood weighting will degrade as the number of evidence variables increases. This is because most samples will have very low weights and the final estimate will be dominated by a tiny fraction of samples with relatively high weights— In other words, a large portion of the samples do not contribute to the estimate anyway.

For large networks it is customary to use MCMC methods. These use a different technique which does not involve generating a sample from scratch as in direct sampling methods.

We shall study a particular version of MCMC called Gibbs sampling.

Gibbs sampling

Gibbs sampling generates an event by making a random change to a preceding event.

It is therefore helpful to think of the network as being in a particular current state (which species an event). The next state is reached by sampling a value for one of the non-evidence variables X_i, conditioned on the current values of all the other variables.

The Gibbs sampler thus wanders randomly in the state space (the space of possible complete assignments), flipping one variable at a time, but keeping the evidence variables fixed.

The idea is that as the sampling settles into a dynamic equilibrium, the long-run fraction of time spent in each state is exactly proportional to its posterior probability.

In other words, given enough sampling steps, the Gibbs sampler generates a sample according to the probability of the event defined by the sample, conditioned on the observed evidence.

Example: Gibbs sampling

Referring to the wet grass network again, lets say we wish to obtain

$$P(Rain|Sprinkler = true, WetGrass = true)$$

The evidence variables $Sprinkler$ and $WetGrass$ are fixed to their observed values.

The non-evidence variables (which include the query variable) $Cloudy$ and $Rain$ are initialised randomly— say, to $true$ and $false$ respectively.

Therefore, the initial state is $[true, true, false, true]$.

Example: Gibbs sampling (cont.)

Next, we sample new values for the non-evidence variables:

- Construct the probability distribution
  $P(Cloudy|Sprinkler = true, Rain = false, WetGrass = true)$
  and sample a new value for $Cloudy$. Suppose this returns $false$. Then the new state is $[false, true, false, true]$.
- Construct the probability distribution
  $P(Rain|Cloudy = false, Sprinkler = true, WetGrass = true)$
  and sample a new value for $Rain$. Suppose this returns $true$. Then the new state is $[false, true, true, true]$.

The above process of cycling through and flipping the unobserved variables are repeated until convergence (i.e., when the changes between states become insignificant).

Example: Gibbs sampling (cont.)

Lets say the process is carried out for 100 iterations, where $Rain = true$ for 57 states and $Rain = false$ for 43 states.

The desired probability estimate is thus

$$\begin{aligned} & \hat{P}(Rain|Sprinkler = true, WetGrass = true) \\ = \; & \alpha\langle 57, 43\rangle = \langle 0.57, 0.43\rangle \end{aligned}$$

Constructing the sampling distributions

In each iteration of Gibbs sampling the new value of a non-evidence variable X_i is sampled from a corresponding sampling distribution.

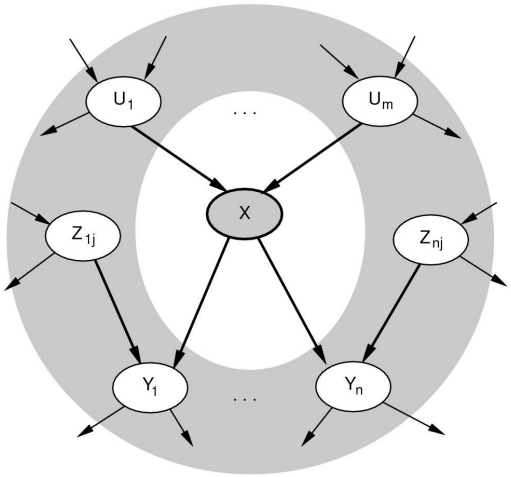
The sampling distribution is defined as

$$P(X_i|\text{values of all other variables})$$

Recall that if the values of the variables in the Markov Blanket of a particular variable are known, that variable is conditionally independent of all other variables.

Constructing the sampling distributions (cont.)

Recall that the Markov Blanket of a variable comprises of the parents, children, and children's parents of the variable.



Constructing the sampling distributions (cont.)

Therefore

$$\begin{aligned} & P(X_i|\text{values of all other variables}) \\ = \; & P(X_i|\text{values of Markov Blanket of } X_i) \\ = \; & \alpha P(X_i|Parents(X_i)) \times \prod_{\forall Y_j \in Children(X_i)} P(Y_j|Parents(Y_j)) \end{aligned}$$

Example: Sampling distributions

Say we wish to construct the sampling distribution for $Cloudy$ given the current state $[true, true, false, true]$.

Then, we obtain

$$\begin{aligned} & P(Cloudy|Sprinkler = true, Rain = false, WetGrass = true) \\ = \; & P(Cloudy|Sprinkler = true, Rain = false) \\ = \; & \alpha P(Cloudy)P(Sprinkler = true|Cloudy)P(Rain = false|Cloudy) \\ = \; & \alpha\langle 0.5 \times 0.1 \times 0.2, 0.5 \times 0.5 \times 0.8\rangle \\ = \; & \alpha\langle 0.01, 0.2\rangle = \langle 0.05, 0.95\rangle \end{aligned}$$

and sample a new value for $Cloudy$ from this distribution.

Now, as exercise construct the sampling distribution for $Rain$ given the current state is $[false, true, false, true]$.