Lecture 7: Approximate Inference: Sampling

- Random sampling from a Bayes net
- Logical (rejection) sampling
- Likelihood weighting
- Gibbs sampling and MCMC

Random sampling

Main idea:

- Use the Bayes net as a model of the world, and generate samples

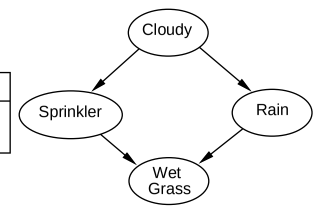
  A sample is a tuple where every random variable is instantiated to some value
- Then approximate the required probability distribution using counts

Two main kinds of methods:

- Forward sampling
- Monte Carlo Markov Chain

Example: Sprinkler network

P(C) = .5



C	P(S)
T	.10
F	.50

C	P(R)
T	.80
F	.20

S	R	P(W)
T	T	.99
T	F	.90
F	T	.90
F	F	.00

Example: Forward sampling

1. Sample C according to its probability distribution. Say $C = 1$.
2. Sample R according to $P(R|C = 1)$. Say $R = 1$.
3. Sample S according to $P(S|C = 1)$. Say $S = 0$.
4. Sample W according to $P(W|R = 1, S = 0)$. Say $W = 1$.

Now we have a complete sample: $\langle C = 1, R = 1, S = 0, W = 1\rangle$

We repeat the steps above to generate a new sample.

E.g. $C = 0, R = 0, S = 1, W = 1$

This process is called **logic sampling**

Example (continued)

Suppose we generate N samples using the above technique.
How do we compute $P(W)$?

$$P(W=1) \approx \frac{n(W=1)}{N}$$

How do we compute $P(W=1|C=1)$?

$$\begin{aligned} P(W=1|C=1) &= \frac{P(C=1,W=1)}{P(C=1)} \\ &\approx \frac{n(C=1,W=1)}{N}\frac{N}{n(C=1)} = \frac{n(C=1,W=1)}{n(C=1)} \end{aligned}$$

Note that we did not use all the samples in this computation!
Only the samples in which $C=1$ were used.

Rejection sampling

- Generate samples by forward sampling of the network:
  - Let $X_1, \ldots X_n$ be an ordering of the variables consistent with the arc direction in the Bayes net structure
  - For $i=1,\ldots,n$, sample X_i from $P(X_i|Parents(X_i))$.

  Note that all the parents of X_i are surely instantiated when we get to sample X_i.
- Throw away the samples inconsistent with the evidence

Problem: If the evidence is unlikely, then we will throw away most samples, and it takes a long time to gather enough data for a reliable estimate.

Becoming more efficient

Suppose we want to estimate $P(W=1|C=1)$. Before, we threw away the samples in which $C=0$. So why generate them in the first place?

Main idea: Fix the values for the evidence variables, sample only the other variables. Then we can use all the samples.

In our case, set $C=1$, then:

1. Sample R from $P(R|C=1)$
2. Sample S from $P(S|C=1)$
3. Sample W from $P(W|R,S)$

Now if we approximate $P(W=1|C=1)$ by $\frac{n(W=1)}{N}$, we should be all set.

Downstream evidence

Suppose we want to compute $P(C|W=1)$. We fix $W=1$ and we need to sample C, R, S.

- We would like to sample R from $P(R|W=1)$.
  But we do not have these probabilities! We could do arc reversal on the network, but that can lead to much larger tables.
- Idea: sample the network top-down like before, but fix the values of the evidence variables. E.g.
  1. Sample C according to $P(C)$. Say $C=0$.
  2. Sample R according to $P(R|C=0)$. Say $R=0$
  3. Sample S according to $P(S|C=0)$. Say $S=0$.
  4. $W=1$ (since it is the evidence)

  But now we generated a sample that has 0 probability!

A simple case

Consider a very simple network: $X \to Y$.

We want to compute $P(X|Y=1)$.

1. Sample X from $P(X)$
2. Set $Y = 1$

Problem: These samples come from $P(X)$, not $P(X, Y=1)$. So we have:

$$\frac{n(X=1, Y=1)}{N} \approx P(X=1), \text{ not } P(X=1, Y=1)$$

A simple case (continued)

To see the fix to this problem, let us consider how we would compute $P(X=1, Y=1)$ exactly:

$$P(X=1, Y=1) = P(Y=1|X=1)P(X=1)$$

Since our sample count approximates $P(X=1)$, all we have to do is multiply the estimate by the **weight** $P(Y=1|X=1)$.

We do the same thing to estimate $P(Y=1, X=0)$. Then we can approximate the conditional as usual.

This is called **likelihood weighting**

Likelihood weighting

Let $X_1, \ldots X_n$ be an ordering of the variables consistent with the arc direction in the Bayes net structure

1. Repeat for $i = 1, \ldots, N$ times:
   (a) $w = 1$
   (b) For $j = 1, \ldots, n$ do:
      - If X_j has been observed (as evidence), $w \leftarrow w \cdot P(X_j = x_j | Parents(X_j))$
      - Else sample X_j from $P(X_j | Parents(X_j))$
2. $P(\mathbf{q}|\mathbf{e}) \approx \frac{\sum_{i=1}^{N} w_i n(\mathbf{q})}{\sum_{i=1}^{N} w_i}$

Importance sampling

Likelihood weighting is a special case of a more general procedure, called **importance sampling**

- Suppose we want to estimate the expected value of a random variable X drawn according to the probability distribution $p(X)$.
- But instead, we have only samples drawn according to $p'(X)$.
- We do a simple trick:

$$E(X) = \sum_i x_i p(X = x_i) = \sum_i x_i p'(X = x_i) \frac{p(X = x_i)}{p'(X = x_i)}$$

- So we will average each sample x_i **weighted** by the ratio of its probability under the target and the sampling distribution.

We will use this idea again in Markov Decision Processes.

Error of likelihood weighting

- Intuitively, the weights reflect the probabilities of the samples. So to get a good approximation, we require a certain “mass”
- Several bounds exist, all specifying the total mass as a function of the error guarantees and the “extremeness” of the CPDs
- Hence, we might still need a lot of samples before we can make good estimates!

13

MCMC methods

Another quite different idea is to generate a “random walk” over variable assignments that are consistent with the evidence.

- View the sampling process as a Markov Chain
- We always generate a new sample by “perturbing” a previously generated sample
- In the limit, if we are careful, the samples will approximate the desired distribution

14

Gibbs sampling

1. Initialization
   - For each evidence variable X_j, set it to the observed value x_j
   - Set all other variables to random values (e.g. by forward sampling)

   This gives us a sample $x_1, \ldots, x_n$.
2. Repeat
   - Pick a variable X_i uniformly randomly
   - Sample x_i' from $P(X_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, \mathbf{e})$.
   - For all other variables, preserve the existing values: $x_j' = x_j, \forall j \neq i$
   - The new sample is $x_1', \ldots, x_n'$

15

Why Gibbs works in Bayes nets

The key step is sampling according to $P(X_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, \mathbf{e})$. But in Bayes nets, we know that: $P(X_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(X_i | MB(X_i))$ where $MB(X_i)$ is the Markov blanket of X_i (parents, children and spouses). So we only need to figure out $P(X_i | MB(X_i))$.

Let $Y_j, j = 1, \ldots, k$ be the children of X_i
We can show (problem set 3) that:

$$P(x_i | MB(X_i)) = \frac{P(x_i | Parents(X_i)) \prod_{j=1}^{k} P(Y_j | Parents(Y_j))}{\sum_{x_i'} P(x_i' | Parents(X_i)) \prod_{j=1}^{k} P(Y_j | Parents(Y_j))}$$

16

Example

1. Generate a first sample: $C = 0, R = 0, S = 0, W = 1$.
2. Pick R, sample it from $P(R|C = 0, W = 1, S = 0)$. Suppose we get $R = 1$.
3. Our new sample is $C = 0, R = 1, S = 0, W = 1$
4. ....

Implementing Gibbs sampling

- Note that the samples we get in the beginning of the sampling are "unlikely". We need to run Gibbs sampling for a while before we start getting "good" samples. This stage is called "burn in"
- Ways of implementing:
  - Run M times starting from different states. Each time, run for N steps, for some fairly large N, then take just one resulting sample. Has a good chance of covering the space of possible samples
  - Start just from one sample, run for a really long time, then take M samples. In this case, the samples will not be independent (but the correlation is weak)
  - A hybrid of the two

Analyzing Gibbs sampling

- Consider the variables $X_1, \ldots, X_n$. Each possible assignment of values to these variables is a state of the world, $\langle x_1, \ldots, x_n \rangle$.
- In Gibbs sampling, we start from a given state $s = \langle x_1, \ldots, x_n \rangle$. Based on this, we generate a new state, $s' = \langle x'_1, \ldots, x'_n \rangle$.
  The new state only depends on the previous state, not on any state that could have happened before!
- For any s, s', there is a well-defined probability of generating s' if we are in s (what is that?)

Gibbs sampling constructs a **Markov chain** over the Bayes net

Markov chains

A Markov chain is defined by:

- A set of states S
- A starting distribution over the set of states $p(s) = P(s_0 = s)$
- A stationary transition probability $p_{ss'} = P(s_{t+1} = s'|s_t = s)$

$$s_0 \rightarrow s_1 \rightarrow \ldots \rightarrow s_t \rightarrow s_{t+1} \rightarrow \ldots$$

Steady-state (stationary) distribution

What is $P(s_t = j | s_0 = i)$?

$$\begin{aligned} P(s_1 = j | s_0 = i) &= p_{ij} \\ P(s_{t+1} = j | s_0 = i) &= \sum_k P(s_{t+1} = j | s_t = k) P(s_t = k | s_0 = i) \\ &= \sum_k p_{kj} P(s_t = k | s_0 = i) \end{aligned}$$

Under reasonable assumptions, this process converges to a unique solution, called the **steady-state distribution**:

$$p^*(i) = \lim_{t \to \infty} P(X_t = i | X_0)$$

Note that $p^*(i)$ does not depend at all on the start state distribution

Sampling the steady-state distribution

The MC theory suggests a way of sampling the stationary distribution:

- Set $X_1 = i$ for some arbitrary i
- For $t = 1, \ldots, M$, if $s_t = s$, sample a value s' for s_{t+1} based on $p_{ss'}$
- Return s_M.

If M is large enough, this will be a sample from p^*

Markov Chain Monte Carlo

- Construct a Markov Chain corresponding to the Bayes net
- Make sure that the chain has the right stationary distribution
- Simulate the chain for N steps to get a sample

Gibbs sampling is the simplest illustration of this idea.

Designing Markov Chains

How do we ensure that the Markov Chain has the “right” probability distribution?

Look again at:

$$p^*(i) = \sum_j p_{ij} p^*(i) = \sum_j p_{ji} p^*(j)$$

If $\frac{p_{ij}}{p_{ji}} = \frac{p^*(j)}{p^*(i)}$, this equality is satisfied.

This gives us a condition that we can check locally!