CS 188: Artificial Intelligence
Spring 2010

Lecture 18: Bayes Nets V
3/30/2010

Pieter Abbeel – UC Berkeley

Many slides over this course adapted from Dan Klein, Stuart Russell, Andrew Moore

---

Announcements

- Midterms
  - In glookup
- Assignments
  - W5 due Thursday
  - W6 going out Thursday
- Midterm course evaluations in your email soon

Outline

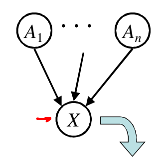
- Bayes net refresher:
  - Representation
  - Inference
    - Enumeration
    - Variable elimination
- Approximate inference through sampling
- Value of information

Bayes' Net Semantics

- A set of nodes, one per variable X
- A directed, acyclic graph
- A conditional distribution for each node
  - A collection of distributions over X, one for each combination of parents' values

    $P(X|a_1 \dots a_n)$

  - CPT: conditional probability table
  - Description of a noisy "causal" process



$P(X|A_1 \dots A_n)$

A Bayes net = Topology (graph) + Local Conditional Probabilities

---

Probabilities in BNs

- For all joint distributions, we have (chain rule):

$$P(x_1, x_2, \dots x_n) = \prod_{i=1}^{n} P(x_i|x_1, \dots, x_{i-1})$$

- Bayes' nets implicitly encode joint distributions
  - As a product of local conditional distributions
  - To see what probability a BN gives to a full assignment, multiply all the relevant conditionals together:

$$P(x_1, x_2, \dots x_n) = \prod_{i=1}^{n} P(x_i|parents(X_i))$$

- This lets us reconstruct any entry of the full joint
- Not every BN can represent every joint distribution
  - The topology enforces certain conditional independencies

Inference by Enumeration

- Given unlimited time, inference in BNs is easy
- Recipe:
  - State the marginal probabilities you need
  - Figure out ALL the atomic probabilities you need
  - Calculate and combine them

$$P(X_5 = x_5 | X_1 = +x_1, X_4 = +x_4) = \frac{P(X_5 = +x_5, X_1 = +x_1, X_4 = +x_4)}{\sum_{x_5} P(X_5 = x_5, X_1 = +x_1, X_4 = +x_4)}$$

- Building the full joint table takes time and space exponential in the number of variables

General Variable Elimination

- Query: $P(Q|E_1 = e_1, \dots E_k = e_k)$
- Start with initial factors:
  - Local CPTs (but instantiated by evidence)
- While there are still hidden variables (not Q or evidence):
  - Pick a hidden variable H
  - Join all factors mentioning H
  - Eliminate (sum out) H
- Join all remaining factors and normalize
- Complexity is exponential in the number of variables appearing in the factors---can depend on ordering but even best ordering is often impractical

Approximate Inference

- Basic idea:
  - Draw N samples from a sampling distribution S
  - Compute an approximate posterior probability
  - Show this converges to the true probability P
- Why sample?
  - Learning: get samples from a distribution you don't know
  - Inference: getting a sample is faster than computing the right answer (e.g. with variable elimination)

with probability $1-\delta$ we have: $|\hat{x} - E[x]| \le \varepsilon$ if for $x \in [0,1]$

number of samples N satisfies

$N \ge \frac{1}{2\varepsilon^2} \log \frac{1}{\delta}$

$\hat{x} = \frac{1}{N} \sum_{i=1}^{N} x^{(i)}$

Prior Sampling

$P(C)$

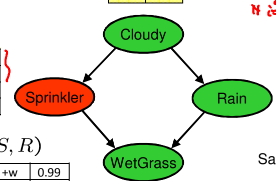
+c	0.5
-c	0.5

$P(S|C)$

+c	+s	0.1
	-s	0.9
-c	+s	0.5
	-s	0.5

$P(R|C)$

+c	+r	0.8
	-r	0.2
-c	+r	0.2
	-r	0.8



$P(W|S,R)$

+s	+r	+w	0.99
		-w	0.01
	-r	+w	0.90
		-w	0.10
-s	+r	+w	0.90
		-w	0.10
	-r	+w	0.01
		-w	0.99

Samples:

+c, -s, +r, +w
-c, +s, -r, +w
…

estimated $P(W=+w) = \frac{2}{2} = 1.0$

Prior Sampling

- This process generates samples with probability:

$$S_{PS}(x_1 \dots x_n) = \prod_{i=1}^{n} P(x_i|\text{Parents}(X_i)) = P(x_1 \dots x_n)$$

…i.e. the BN's joint probability

- Let the number of samples of an event be $N_{PS}(x_1 \dots x_n)$
- Then

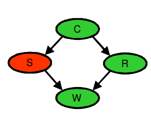
$$\begin{aligned} \lim_{N\to\infty} \hat{P}(x_1, \dots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \dots, x_n)/N \\ &= S_{PS}(x_1, \dots, x_n) \\ &= P(x_1 \dots x_n) \end{aligned}$$

- I.e., the sampling procedure is consistent

Example

- We'll get a bunch of samples from the BN:

+c, -s, +r, +w
+c, +s, +r, +w
-c, +s, +r, -w
+c, -s, +r, +w
-c, -s, -r, +w



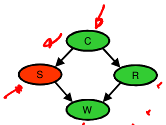
- If we want to know P(W)
  - We have counts <+w:4, -w:1>
  - Normalize to get P(W) = <+w:0.8, -w:0.2>
  - This will get closer to the true distribution with more samples
  - Can estimate anything else, too
  - What about P(C| +w)? P(C| +r, +w)? P(C| -r, -w)?
  - Fast: can use fewer samples if less time (what's the drawback?)

$P(c|+w) = \frac{3}{4}$
$P(\neg c|+w) = \frac{1}{4}$

$P(-r,-w) \le \varepsilon$

Rejection Sampling

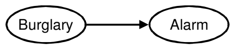
- Let's say we want P(C)
  - No point keeping all samples around
  - Just tally counts of C as we go
- Let's say we want P(C| +s)
  - Same thing: tally C outcomes, but ignore (reject) samples which don't have S=+s
  - This is called rejection sampling
  - It is also consistent for conditional probabilities (i.e., correct in the limit)



+c, -s, +r, +w
+c, +s, +r, +w
-c, +s, +r, -w
+c, -s, +r, +w
-c, -s, -r, +w

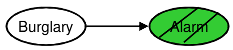
Likelihood Weighting

- Problem with rejection sampling:
  - If evidence is unlikely, you reject a lot of samples
  - You don't exploit your evidence as you sample
  - Consider P(B|+a)



-b, -a
-b, -a
-b, -a
-b, -a
+b, +a

- Idea: fix evidence variables and sample the rest



-b +a P(+a|-b)
-b, +a P(+a|-b)
-b, +a P(+a|-b)
-b, +a P(+a|-b)
+b, +a P(+a|+b)

- Problem: sample distribution not consistent!
- Solution: weight by probability of evidence given parents

Likelihood Weighting

$P(C)$

+c	0.5
-c	0.5

$P(S|C)$

+c	+s	0.1
	-s	0.9
-c	+s	0.5
	-s	0.5

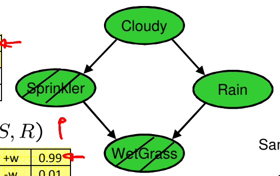
$P(R|C)$

+c	+r	0.8
	-r	0.2
-c	+r	0.2
	-r	0.8

$P(W|S,R)$

+s	+r	+w	0.99
		-w	0.01
	-r	+w	0.90
		-w	0.10
-s	+r	+w	0.90
		-w	0.10
	-r	+w	0.01
		-w	0.99

| +s, +w)



Samples:

+c, +s, +r, +w

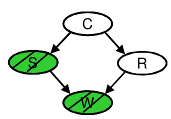
…

$w = 1.0 \times 0.1 \times 0.99$

Likelihood Weighting

- Sampling distribution if z sampled and e fixed evidence

$$S_{WS}(\mathrm{z}, \mathrm{e}) = \prod_{i=1}^{l} P(z_i|\mathrm{Parents}(Z_i))$$



- Now, samples have weights

$$w(\mathrm{z}, \mathrm{e}) = \prod_{i=1}^{m} P(e_i|\mathrm{Parents}(E_i))$$

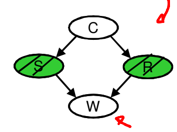
- Together, weighted sampling distribution is consistent

$$S_{\mathrm{WS}}(z, e) \cdot w(z, e) = \prod_{i=1}^{l} P(z_i|\mathrm{Parents}(z_i)) \prod_{i=1}^{m} P(e_i|\mathrm{Parents}(e_i))$$

$$P(z, e) = P(\mathrm{z}, \mathrm{e})$$

Likelihood Weighting

- Likelihood weighting is good
  - We have taken evidence into account as we generate the sample
  - E.g. here, W's value will get picked based on the evidence values of S, R
  - More of our samples will reflect the state of the world suggested by the evidence
- Likelihood weighting doesn't solve all our problems
  - Evidence influences the choice of downstream variables, but not upstream ones (C isn't more likely to get a value matching the evidence)
- We would like to consider evidence when we sample every variable



Markov Chain Monte Carlo*

- *Idea*: instead of sampling from scratch, create samples that are each like the last one.
- *Procedure*: resample one variable at a time, conditioned on all the rest, but keep evidence fixed. E.g., for P(b|c):

$+b \to +a \to +c \qquad -b \to +a \to +c \qquad -b \to -a \to +c$

- *Properties*: Now samples are not independent (in fact they're nearly identical), but sample averages are still consistent estimators!

P(A|+b) * P(+c|A)

P(C|+a)

- *What's the point*: both upstream and downstream variables condition on evidence.

Inference

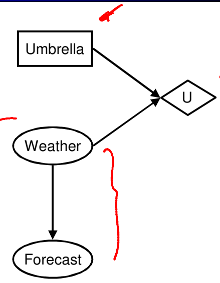
CS 281A (Fall, Jordan, Wainwright)

Exact
- Enumeration
- Variable elimination — Sum product / Junction tree (Clique tree) +

Approx.
- Sampling { forward, rejection, likelihood weighting, MCMC (Gibbs sampling) * }
- Variational methods *

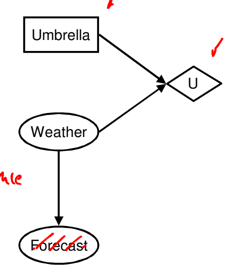
Decision Networks

- MEU: choose the action which maximizes the expected utility given the evidence
- Can directly operationalize this with decision networks
  - Bayes nets with nodes for utility and actions
  - Lets us calculate the expected utility for each action
- New node types:
  - Chance nodes (just like BNs)
  - Actions (rectangles, cannot have parents, act as observed evidence)
  - Utility node (diamond, depends on action and chance nodes)



Decision Networks

- Action selection:
  - Instantiate all evidence
  - Set action node(s) each possible way
  - Calculate posterior for all parents of utility node, given the evidence = inference
  - Calculate expected utility for each action
  - Choose maximizing action



Example: Decision Networks

Umbrella = leave

$$\mathrm{EU}(\mathrm{leave}) = \sum_w P(w)U(\mathrm{leave}, w)$$

$$= 0.7 \cdot 100 + 0.3 \cdot 0 = 70$$

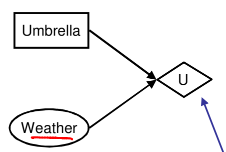
Umbrella = take

$$\mathrm{EU}(\mathrm{take}) = \sum_w P(w)U(\mathrm{take}, w)$$

$$= 0.7 \cdot 20 + 0.3 \cdot 70 = 35$$

Optimal decision = leave

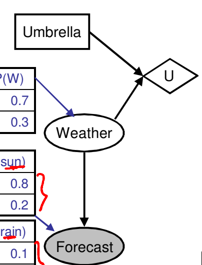
$$\mathrm{MEU}(\phi) = \max_a \mathrm{EU}(a) = 70$$



W	P(W)
sun	0.7
rain	0.3

A	W	U(A,W)
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70

Evidence in Decision Networks



W	P(W)
sun	0.7
rain	0.3

F	P(F|sun)
good	0.8
bad	0.2

F	P(F|rain)
good	0.1
bad	0.9

- Find P(W|F=bad)
  - Select for evidence

W	P(W)
sun	0.7
rain	0.3

$P(W)$

W	P(F=bad|W)
sun	0.2
rain	0.9

$P(bad|W)$

  - First we join P(W) and P(bad|W)
  - Then we normalize

W	P(W,F=bad)
sun	0.14
rain	0.27

$P(W, bad)$

W	P(W | F=bad)
sun	0.34
rain	0.66

$P(W|F = \mathrm{bad})$

Example: Decision Networks

Umbrella = leave

$$\mathrm{EU}(\mathrm{leave}|\mathrm{bad}) = \sum_w P(w|\mathrm{bad})U(\mathrm{leave}, w)$$

$$= 0.34 \cdot 100 + 0.66 \cdot 0 = 34$$

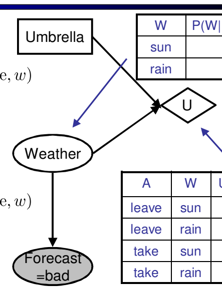
Umbrella = take

$$\mathrm{EU}(\mathrm{take}|\mathrm{bad}) = \sum_w P(w|\mathrm{bad})U(\mathrm{take}, w)$$

$$= 0.34 \cdot 20 + 0.66 \cdot 70 = 53$$

Optimal decision = take

$$\mathrm{MEU}(F = \mathrm{bad}) = \max_a \mathrm{EU}(a|\mathrm{bad}) = 53$$



W	P(W|F=bad)
sun	0.34
rain	0.66

A	W	U(A,W)
leave	sun	100
leave	rain	0
take	sun	20
take	rain	70