Bayes Networks

6.872/HST.950
What Probabilistic Models Should We Use?

- Full joint distribution
  - Completely expressive
  - Hugely data-hungry
  - Exponential computational complexity
- Naive Bayes (full conditional independence)
  - Relatively concise
    - Need data $\sim (#\text{hypotheses}) \times (#\text{features}) \times (#\text{feature-vals})$
    - Fast $\sim (#\text{features})$
  - Cannot express dependencies among features or among hypotheses
  - Cannot consider possibility of multiple hypotheses co-occurring
Bayesian Networks
(aka Belief Networks)

- Graphical representation of dependencies among a set of random variables
  - Nodes: variables
  - Directed links to a node from its parents: direct probabilistic dependencies
  - Each $X_i$ has a conditional probability distribution, $P(X_i|\text{Parents}(X_i))$, showing the effects of the parents on the node.
  - The graph is directed (DAG); hence, no cycles.
- This is a language that can express dependencies between Naive Bayes and the full joint distribution, more concisely
  - Given some new evidence, how does this affect the probability of some other node(s)? $P(X|E)$ —belief propagation/updating
  - Given some evidence, what are the most likely values of other variables? $\arg\max_X P(X|E)$ —MAP explanation
Burglary Network
(due to J. Pearl)

- Burglary
- Earthquake
- Alarm
- JohnCalls
- MaryCalls
Burglary Network
(due to J. Pearl)

P(B) = 0.001

P(E) = 0.002

| B | E | P(A|B,E) |
|---|---|---------|
| t | t | 0.95    |
| t | f | 0.94    |
| f | t | 0.29    |
| f | f | 0.001   |

P(M|A) =

| A | P(M|A) |
|---|--------|
| t | 0.70   |
| f | 0.01   |

P(J|A) =

| A | P(J|A) |
|---|--------|
| t | 0.90   |
| f | 0.05   |
If everything depends on everything

- This model requires just as many parameters as the full joint distribution!
Computing the Joint Distribution from a Bayes Network

• As usual, we abuse notation:

\[ P(X_1 = x_1 \land \ldots \land X_n = x_n) \text{ is written as } P(x_1, \ldots, x_n) \]

• \[ P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i | \text{Par}(X_i)) \]

• E.g., what’s the probability that an alarm has sounded, there was neither an earthquake nor a burglary, and both John and Mary called?

\[ P(j \land m \land a \land \neg b \land \neg e) \]

\[ = P(J|a)P(m|a)P(a|\neg b \land \neg e)P(\neg e)P(\neg b) \]

\[ = 0.90 \times 0.70 \times 0.001 \times 0.999 \times 0.998 = 0.00062 \]
Requirements for Constructing a BN

• Recall that the definition of the conditional probability was
  \[ P(x|y) = P(x \wedge y)/P(y) \]
  and thus we get the chain rule,
  \[ P(x \wedge y) = P(x|y)P(y) \]

• Generalizing to \( n \) variables,
  \[ P(x_1, \ldots, x_n) = P(x_n|x_{n-1}, \ldots, x_1)P(x_{n-1}, \ldots, x_1) \]
  and repeatedly applying this idea,

  \[
P(x_1, \ldots, x_n) = P(x_n|x_{n-1}, \ldots, x_1)P(x_{n-1}|x_{n-2}, \ldots, x_1) \cdots P(x_2|x_1)P(x_1)
  = \prod_{i=1}^{n} P(x_i|x_{i-1}, \ldots, x_1)
  = \prod_{i=1}^{n} P(x_i|\text{Par}(x_i))
  \]

• This “works” just in case we can define a partial order so that
  \[ \text{Par}(X_i) \subseteq \{X_{i-1}, \ldots, X_1\} \]
Topological Interpretations

A node, $X$, is conditionally independent of its non-descendants, $Z_i$, given its parents, $U_i$.

A node, $X$, is conditionally independent of all other nodes in the network given its Markov blanket: its parents, $U_i$, children, $Y_i$, and children’s parents, $Z_i$. 
BN’s can be Compact

• For a network of 40 binary variables, the full joint distribution has $2^{40}$ entries (> 1,000,000,000,000)
• If $|\text{Par}(x_i)| \leq 5$, however, then the 40 (conditional) probability tables each have $\leq 32$ entries, so the total number of parameters $\leq 1,280$
• Largest medical BN I know (Pathfinder) had 109 variables! $2^{109} \approx 10^{36}$
How *Not* to Build BN’s

- With the wrong ordering of nodes, the network becomes more complicated, and requires more (and more difficult) conditional probability assessments.
Simplifying Conditional Probability Tables

• Do we know any structure in the way that Par(x) “cause” x?
• If each destroyer can sink the ship with probability $P(s|d_i)$, what is the probability that the ship will sink if it’s attacked by both?
  $$(1 - P(s|d_1, d_2)) = (1 - P(s|d_1))(1 - P(s|d_2)) (1 - l)$$
• For $|\text{Par}(x)| = n$, this requires $O(n)$ parameters, not $O(k^n)$
Inference

• Recall the two basic inference problems: Belief propagation & MAP explanation
• Trivially, we can enumerate all “matching” rows of the joint probability distribution
• For poly-trees (not even undirected loops—i.e., only one connection between any pair of nodes; like our Burglary example), there are efficient linear algorithms, similar to constraint propagation
• For arbitrary BN’s, all inference is NP-hard!
  • Exact solutions
  • Approximation
Exact Solution of BN’s (Burglary example)

\[
P(b|j, m) = \alpha \sum_e \sum_a P(b)P(e)P(a|b, e)P(j|a)P(m|a)
\]

\[
= \alpha P(b) \sum_e P(e) \sum_a P(a|b, e)P(j|a)P(m|a)
\]

\[
P(B|j, m) = \alpha \{0.00059224, 0.0014919\} \approx \{0.28, 0.72\}
\]

- **Notes:**
  - Sum over all “don’t care” variables
  - Factor common terms out of summation
  - Calculation becomes a sum of products of sums of products ...
Poly-trees are easy

- Singly-connected structures allow propagation of observations via single paths
- “Down” is just use of conditional probability
- “Up” is just Bayes rule
- Formulated as message propagation rules
- Linear time (network diameter)
- Fails on general networks!
Exact Solution of BN’s
(non-poly-trees)

\[ P(A, B, C, D, E) = P(A)P(B|A)P(C|A)P(D|B, C)P(E|C) \]

- What is the probability of a specific state, say \( A=t, B=f, C=t, D=t, E=f \)?
  \[ p(a, \neg b, c, d, \neg e) = p(a), p(\neg b|a)p(c, a)p(d|\neg b, c)p(\neg e|c) \]

- What is the probability that \( E=t \) given \( B=t \)?
  \[ p(e|b) = p(e, b)/p(b) \]

- Consider the term \( P(e, b) \)

\[
P(e, b) = \sum_{A,C,D} P(A, b, C, D, e)
\]

\[
= \sum_{A,C,D} P(A)P(b|A)P(C|A)P(D|b, C)P(e|C)
\]

\[
= \sum_{C} P(e|C) \left( \sum_{A} P(A)P(C|A)P(b|A) \right) \left( \sum_{D} P(D|b, C) \right)
\]

Alas, optimal factoring is NP-hard

- 12 instead of 32 multiplications (even in this small example)
Other Exact Methods

- **Join-tree**: Merge variables into (small!) sets of variables to make graph into a poly-tree. Most commonly-used; aka *Clustering, Junction-tree, Potential*

- **Cutset-conditioning**: Instantiate a (small!) set of variables, then solve each residual problem, and add solutions weighted by probabilities of the instantiated variables having those values

- ...  

- All these methods are essentially equivalent; with some time-space tradeoffs.
Approximate Inference in BN’s

- Direct Sampling—samples joint distribution
- Rejection Sampling—computes $P(\mathbf{X} | e)$, uses ancestor evidence nodes in sampling
- Likelihood Weighting—like Rejection Sampling, but weights by probability of descendant evidence nodes
- Markov chain Monte Carlo
  - Gibbs and other similar sampling methods
function Prior-Sample(bn) returns an event sampled from bn
input: bn, a Bayes net specifying the joint distribution $P(X_1, ... X_n)$
x := an event with $n$ elements
for $i = 1$ to $n$ do
  $x_i$ := a random sample from $P(X_i|\text{Par}(X_i))$
return $x$

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

• From a large number of samples, we can estimate all joint probabilities
• The probability of an event is the fraction of all complete events generated by PS that match the partially specified event
• hence we can compute all conditionals, etc.
Rejection Sampling

function Rejection-Sample(X, e, bn, N) returns an estimate of P(X|e)
inputs:
bn, a Bayes net
X, the query variable
e, evidence specified as an event
N, the number of samples to be generated
local: K, a vector of counts over values of X, initially 0

for j = 1 to N do
    y := PriorSample(bn)
    if y is consistent with e then
        K[v] := K[v]+1 where v is the value of X in y
return Normalize(K[X])

• Uses PriorSample to estimate the proportion of times each value of X appears in samples that are consistent with e
• But, most samples may be irrelevant to a specific query, so this is quite inefficient
Likelihood Weighting

- In trying to compute $\Pr(X|\mathbf{e})$, where $\mathbf{e}$ is the evidence (variables with known, observed values),
  - Sample only the variables other than those in $\mathbf{e}$
  - Weight each sample by how well it predicts $\mathbf{e}$

\[
S_{\text{WS}}(z,e)w(z,e) = \prod_{i=1}^{l} P(z_i|\text{Par}(Z_i)) \prod_{i=1}^{m} P(e_i|\text{Par}(E_i)) = \Pr(z,e)
\]
Likelihood Weighting

\[
S_{WS}(z, e)w(z, e) = \prod_{i=1}^{l} P(z_i | \text{Par}(Z_i)) \prod_{i=1}^{m} P(e_i | \text{Par}(E_i)) \\
= P(z, e)
\]

function Likelihood-Weighting(X, e, bn, N) returns an estimate of $P(X|e)$
inputs: bn, a Bayes net
X, the query variable
e, evidence specified as an event
N, the number of samples to be generated
local: W, a vector of weighted counts over values of X, initially 0
for j = 1 to N do
    y, w := WeightedSample(bn, e)
    if y is consistent with e then
        W[v] := W[v] + w where v is the value of X in y
return Normalize(W[X])

function Weighted-Sample(bn, e) returns an event and a weight
x := an event with n elements; w := 1
for i = 1 to n do
    if $X_i$ has a value $x_i$ in e
        then w := w * $P(X_i = x_i | \text{Par}(X_i))$
        else $x_i$ := a random sample from $P(X_i | \text{Par}(X_i))$
return x, w
function MCMC(X, e, bn, N) returns an estimate of P(X|e)
local: K[X], a vector of counts over values of X, initially 0
Z, the non-evidence variables in bn (includes X)
\(x\), the current state of the network, initially a copy of e
initialize \(x\) with random values for the vars in Z
for \(j = 1\) to \(N\) do
    for each Zi in Z do
        sample the value of Zi in x from P(Zi|mb(Zi)), given the values of mb(Zi) in x
        \(K[v] := K[v]+1\) where v is the value of X in x
return Normalize(K[X])

- Wander incrementally from the last state sampled, instead of re-generating a completely new sample
- For every unobserved variable, choose a new value according to its probability given the values of vars in it Markov blanket (remember, it’s independent of all other vars)
- After each change, tally the sample for its value of X; this will only change sometimes
- Problem: “narrow passages”
Most Probable Explanation

- So far, we have been solving for $P(\mathbf{X}|e)$, which yields a distribution over all possible values of the $x$'s.
- What if we want the best explanation of a set of evidence, i.e., the highest-probability set of values for the $x$'s, given $e$?
- Just maximize over the “don’t care” variables rather than summing.
- This is not necessarily the same as just choosing the value of each $x$ with the highest probability.
Rules and Probabilities

- Many have wanted to put a probability on assertions and on rules, and compute with likelihoods
- E.g., Mycin’s *certainty factor* framework
  - A (p=.3) & B (p=.7) $$\Rightarrow$$ p=.8$$\Rightarrow$$ C (p=?)
- Problems:
  - How to combine uncertainties of preconditions and of rule
  - How to combine evidence from multiple rules
- Theorem: There is NO such algebra that works when rules are considered independently.
- Need BN for a consistent model of probabilistic inference