Announcements

• Midterms graded
• Assignment 2 graded

Mid-term

- Bayes rule:
  \[ P(x|y) = \frac{P(y|x)P(x)}{P(y)} \]
- \( P(x|y) \) = Posterior
- \( P(y|x) \) = Likelihood
- \( P(x) \) = Prior
- \( P(y) \) = Evidence

Inference

- Inference: calculating some statistic from a joint probability distribution
- Examples:
  - Posterior probability:
    \[ P(Q|E_1 = e_1, \ldots, E_k = e_k) \]
  - Most likely explanation:
    \[ \arg\max_q P(Q = q|E_1 = e_1) \]

Reminder: Alarm Network

- 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
- Example:
  \[ P(b|j, m) = \frac{P(b, j, m)}{P(j, m)} \]
Example

\[ P(b_j, m) = \frac{P(b, j, m)}{P(j, m)} \]

\[ P(b, j, m) = P(b, e, a, j, m) + P(b, \bar{e}, a, j, m) + P(b, e, \bar{a}, j, m) + P(b, \bar{e}, \bar{a}, j, m) \]

\[ = \sum_{e, a} P(b, e, a, j, m) \]

We didn’t!

Example

- In this simple method, we only need the BN to synthesize the joint entries

\[ P(b, j, m) = P(b)P(e)P(a|b, e)P(j|a)P(m|a) + P(b)P(\bar{a}|b, e)P(j|\bar{a})P(m|\bar{a}) + P(b)P(\bar{e})P(a|b, \bar{e})P(j|a)P(m|a) + P(b)P(\bar{e})P(\bar{a}|b, \bar{e})P(j|\bar{a})P(m|\bar{a}) \]

Normalization Trick

\[ P(B_j, m) = \frac{P(B, j, m)}{P(j, m)} \]

\[ P(b, j, m) = \sum_{e, a} P(b, e, a, j, m) \]

\[ P(b, j, m) = \sum_{e, a} P(b, e, a, j, m) \]

\[ \left( \frac{P(b, j, m)}{P(b, j, m)} \right) \]

Normalize \[ \left( \frac{P(b, j, m)}{P(b, j, m)} \right) \]

Inference by Enumeration?

Variable Elimination

- Why is inference by enumeration so slow?
  - You join up the whole joint distribution before you sum out the hidden variables
  - You end up repeating a lot of work!

- Idea: interleave joining and marginalizing!
  - Called "Variable Elimination"
  - Still NP-hard, but usually much faster than inference by enumeration

- We’ll need some new notation to define VE

Factor Zoo I

- Joint distribution: P(X,Y)
  - Entries P(x,y) for all x, y
  - Sums to 1

- Selected joint: P(x,Y)
  - A slice of the joint distribution
  - Entries P(x,y) for fixed x, all y
  - Sums to P(x)
Factor Zoo II

- Family of conditionals: P(X | Y)
  - Multiple conditionals
  - Entries P(x | y) for all x, y
  - Sums to |Y|

- Single conditional: P(Y | x)
  - Entries P(y | x) for fixed x, all y
  - Sums to 1

<table>
<thead>
<tr>
<th></th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>cold</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>sun</td>
<td>0.2</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Factor Zoo III

- Specified family: P(Y | X)
  - Entries P(y | x) for fixed y, all x
  - Sums to ... who knows!

<table>
<thead>
<tr>
<th></th>
<th>W</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>cold</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>hot</td>
<td>0.8</td>
<td>0.2</td>
</tr>
</tbody>
</table>

Basic Objects

- Track objects called factors
- Initial factors are local CPT’s
  - One per node in the BN
  - P(B) P(E) P(J|A) P(M|A) P(A|B,E)
- Any known values are specified
  - E.g., if we know J = j and E = ¬e, the initial factors are
  - P(B) P(¬e) P(j|A) P(M|A) P(A|B,¬e)
- VE: Alternately join and marginalize factors

Basic Operation: Join

- First basic operation: join factors
- Combining two factors:
  - Just like a database join
  - Build a factor over the union of the variables involved
- Example:
  \[ P(A|B) \times P(B|C) \rightarrow P(A,B|C) \]
  - Computation for each entry: pointwise products
  \[ \forall a, b, c : P(a,b|c) = P(a|b) \cdot P(b|c) \]

Basic Operation: Eliminate

- Second basic operation: marginalization
- Take a factor and sum out a variable
  - Shrinks a factor to a smaller one
  - A projection operation
- Example:
  \[ P(A, b|C) \rightarrow P(b|C) \]
  - Definition:
  \[ \forall c : P(b|c) = \sum_{a} P(a,b|c) \]
**General Variable Elimination**

- **Query:** \( P(Q | E_1 = e_1, \ldots, E_k = e_k) \)
- **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
  - Project out H
- **Join all remaining factors and normalize**

---

**Example**

\[
\begin{align*}
P(B, j, m) \propto P(B, j, m) \\
P(B) & \quad P(E) & \quad P(j, m | B, E) \\
\end{align*}
\]

Choose A

\[
\begin{align*}
P(A | B, E) \\
P(j | A) & \quad P(j, m, A | B, E) & \quad P(j, m | B, E) \\
P(m | A) & \\
\end{align*}
\]

\[
\begin{align*}
P(B) & \quad P(E) & \quad P(j, m | B, E) \\
\end{align*}
\]

**Variable Elimination**

- **What you need to know:**
  - Should be able to run it on small examples, understand the factor creation / reduction flow
  - Better than enumeration: VE caches intermediate computations
  - Saves time by marginalizing variables as soon as possible rather than at the end
  - Polynomial time for tree-structured graphs – sound familiar?
- **We will see special cases of VE later**
- **Approximations**
  - Exact inference is slow, especially with a lot of hidden nodes
  - Approximate methods give you a (close, wrong?) answer, faster

---

**Sampling**

- **Basic idea:**
  - Draw N samples from a sampling distribution S
  - Compute an approximate posterior probability
  - Show this converges to the true probability \( P \)
- **Outline:**
  - Sampling from an empty network
  - Rejection sampling: reject samples disagreeing with evidence
  - Likelihood weighting: use evidence to weight samples

---

**Prior Sampling**

\[
\begin{align*}
P(C) & \quad P(R | C) \\
\end{align*}
\]

\[
\begin{align*}
C & | P(S, C) \\
T & 10 \\
F & 50 \\
\end{align*}
\]

\[
\begin{align*}
Sprinkle & \quad Rain & \quad C | P(R | C) \\
\end{align*}
\]

\[
\begin{align*}
S & | P(W | S, R) \\
T & T & 99 \\
F & T & 90 \\
F & F & 90 \\
\end{align*}
\]
**Prior Sampling**

- This process generates samples with probability

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

...i.e. the BN's joint probability

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

- Then

\[ \lim_{N \to \infty} P(x_1 \ldots x_n) = \lim_{N \to \infty} N_{PS}(x_1 \ldots x_n)/N = S_P(x_1 \ldots x_n) \]

- I.e., the sampling procedure is consistent

**Example**

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

<table>
<thead>
<tr>
<th>C</th>
<th>¬C</th>
<th>S</th>
<th>¬S</th>
<th>R</th>
<th>¬R</th>
<th>W</th>
<th>¬W</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
</tbody>
</table>

- 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| ¬r)?  P(C| ¬r, ¬w)?

**Rejection Sampling**

- Let’s say we want P(C)

- No point keeping all samples around

- Just tally counts of C outcomes

- 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 rejection sampling

- It is also consistent (correct in the limit)

**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)

- Idea: fix evidence variables and sample the rest

- Problem: sample distribution not consistent!

- Solution: weight by probability of evidence given parents

**Likelihood Sampling**

- Problem with likelihood sampling:

- Fix evidence variables and sample the rest

\[ S_W(z, e) = \prod_{i=1}^{l} P(z_i|\text{Parents}(Z_i)) \]

- Now, samples have weights

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

- Together, weighted sampling distribution is consistent

\[ S_{W}(z, e)w(z, e) = \prod_{i=1}^{m} P(e_i|\text{Parents}(E_i)) \prod_{i=1}^{m} P(e_i|\text{Parents}(E_i)) \]

\[ = P(z, e) \]
### Likelihood Weighting

- Note that likelihood weighting doesn’t solve all our problems
- Rare evidence is taken into account for downstream variables, but not upstream ones
- A better solution is Markov-chain Monte Carlo (MCMC), more advanced
- We’ll return to sampling for robot localization and tracking in dynamic BNs

### History of Sampling Methods

- Monte Carlo methods developed at Los Alamos during the Manhattan project
  - Used to design H-bomb
  - Totally revolutionized statistics from 1990’s on
    - Require fast computers & what else?