CSE 473: Artificial Intelligence
Bayes’ Nets: Inference

Dieter Fox

These slides were created by Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley. All CS188 materials are available at http://ai.berkeley.edu

Bayes’ Nets

- Representation
- Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)
  - Variable elimination (exact, worst-case exponential complexity, often better)
  - Probabilistic inference is NP-complete
  - Sampling (approximate)
- Learning Bayes’ Nets from Data

Inference

- Inference: calculating some useful quantity from a joint probability distribution

Examples:

- Posterior probability
  \[ P(Q | E_1, \ldots, E_k) \]
- Most likely explanation:
  \[ \text{argmax}_Q P(Q | E_1, \ldots, E_k) \]

Inference by Enumeration

- General case:
  - Evidence variables: \( E_1, E_2, \ldots, E_k \)
  - Query variable: \( Q \)
  - Hidden variables: \( H_1, H_2, \ldots, H_l \)
  - All variables

- We want:
  \[ P(Q | E_1, \ldots, E_k) = \frac{1}{Z} \sum_{e_1, \ldots, e_k} P(Q, e_1, \ldots, e_k) \]

- Step 1: Select the entries consistent with the evidence
- Step 2: Sum out \( H \) to get joint of \( Q \) and \( e \)
- Step 3: Normalize

Inference by Enumeration in Bayes’ Net

- Given unlimited time, inference in BNs is easy
- Reminder of inference by enumeration by example:

\[
P(B \mid j, m) \propto P(B, j, m) \]

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

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

\[
= P(B) P(e) P(a \mid B, e) P(j \mid a) P(m \mid a) + P(B) P(e) P(\neg a \mid B, e) P(j \mid a) P(m \mid a) + P(\neg B) P(e) P(a \mid \neg B, e) P(j \mid a) P(m \mid a) + P(\neg B) P(e) P(\neg a \mid \neg B, e) P(j \mid a) P(m \mid a)
\]
Inference by Enumeration?

Why is inference by enumeration so slow?
- You join up the whole joint distribution before you sum out the hidden variables

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

First we’ll need some new notation: factors

Factor Zoo

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 \( P(x) \)

Number of capitals = dimensionality of the table

Factor Zoo I

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

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

Factor Zoo II

Factor Zoo III

Specified family: \( P(Y \mid X) \)
- Entries \( P(y \mid x) \) for fixed \( y \), but for all \( x \)
- Sums to … who knows!

First we’ll need some new notation: factors
Inference by Enumeration: Procedural Outline

- Track objects called factors
- Initial factors are local CPTs (one per node)
- Any known values are selected
  - E.g., if we know \( L = \text{false} \), the initial factors are
  \[
  P(R) \quad P(T|R) \quad P(U|T) \quad P(V|U)
  \]
- Procedure: Join all factors, then eliminate all hidden variables

Example: Traffic Domain

- Random Variables
  - R: Raining
  - T: Traffic
  - L: Late for class!

\[
P(L) = \sum_{r,t} P(r,t,L)
\]
\[
P(L) = \sum_{r,t} P(r)P(t|r)P(L|r,t)
\]

Operation 1: Join Factors

- First basic operation: joining factors
  - Combining factors
    - Join like a database join
    - Get all factors over the joining variable
    - Build a new factor over the union of the variables involved
  - Example: Join on \( R \)

\[
P(R) \times P(T|R) \rightarrow P(R,T)
\]

- Computation for each entry: pointwise products

Example: Multiple Joins

- Example: Join on \( R \)

\[
P(R) \times P(T|R) \times P(U|T) \rightarrow P(R,T,U)
\]
Operation 2: Eliminate

- Second basic operation: marginalization
- Take a factor and sum out a variable
  - Shrinks a factor to a smaller one
  - A projection operation
- Example:

\[
P(R, T') = \sum R\ P(T')
\]

Multiple Elimination

\[
P(R, T, L) \rightarrow \sum \text{out } R\ P(T, L) \rightarrow \sum \text{out } T\ P(L)
\]

Thus Far: Multiple Join, Multiple Eliminate (= Inference by Enumeration)

Marginalizing Early (= Variable Elimination)

Traffic Domain

\[
P(L) = ?
\]

- Inference by Enumeration
- Variable Elimination

Marginalizing Early! (aka VE)
Evidence

- If evidence, start with factors that select that evidence
  - No evidence uses these initial factors:
    \[
    P(R) \quad P(T) \quad P(UT)
    \]
- Computing \(P([+r] + r)\) the initial factors become:
  \[
  P([+r]) \quad P([r] + r) \quad P([UT])
  \]
- We eliminate all vars other than query + evidence

Evidence II

- Result will be a selected joint of query and evidence
  - E.g., for \(P[L|+r]\), we would end up with:
    \[
    P(+r, L) \quad \rightarrow \quad P(L|+r)
    \]
- To get our answer, just normalize this!
  - That’s it!

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
  - Eliminate (sum out) H
- Join all remaining factors and normalize

Example

\[
P(B) \times P(E) \times P(j, m|B, E)
\]

Choose A

\[
P(j, m|A, B, E) \quad P(j, m|B, E)
\]

Finish with B

\[
P(B) \times P(j, m|B)
\]

Example

\[
P(B) \times P(E) \times P(j, m|B, E)
\]

Choose A

\[
P(A|B, E) \quad P(A|B, E) \quad P(j|A) \quad P(m|A)
\]

Same Example in Equations

\[
P(B, j, m) \times P(B, j, m^*)
\]

Choose A

\[
P(j) \times P(E) \times P(j, m|B, E)
\]

Finish with B

\[
P(B) \times P(j, m|B)
\]
Another Variable Elimination Example

Query: \( P(X_3|Y_1 = y_1, Y_2 = y_2, Y_3 = y_3) \)

Start by listing nodes, which gives the following initial factors:

\[
\begin{align*}
&f_1(Y_1) = \sum_{X_1} f(X_1, Y_1, Y_2, Y_3) \\
&f_2(Y_2) = \sum_{X_2} f(X_1, Y_1, Y_2, Y_3) \\
&f_3(Y_3) = \sum_{X_3} f(X_1, Y_1, Y_2, Y_3)
\end{align*}
\]

Eliminate \( X_1 \): this introduces the factor \( \sum_{X_1} f(X_1, Y_1, Y_2, Y_3) \) and we are left with:

\[
\begin{align*}
&f_1(Y_1) = \sum_{X_1} f(X_1, Y_1, Y_2, Y_3) \\
&f_2(Y_2) = \sum_{X_2} f(X_1, Y_1, Y_2, Y_3) \\
&f_3(Y_3) = \sum_{X_3} f(X_1, Y_1, Y_2, Y_3)
\end{align*}
\]

Eliminate \( X_2 \): this introduces the factor \( \sum_{X_2} f(X_1, Y_1, Y_2, Y_3) \) and we are left with:

\[
\begin{align*}
&f_1(Y_1) = \sum_{X_1} f(X_1, Y_1, Y_2, Y_3) \\
&f_2(Y_2) = \sum_{X_2} f(X_1, Y_1, Y_2, Y_3) \\
&f_3(Y_3) = \sum_{X_3} f(X_1, Y_1, Y_2, Y_3)
\end{align*}
\]

Eliminate \( X_3 \): this introduces the factor \( \sum_{X_3} f(X_1, Y_1, Y_2, Y_3) \) and we are left with:

\[
\begin{align*}
&f_1(Y_1) = \sum_{X_1} f(X_1, Y_1, Y_2, Y_3) \\
&f_2(Y_2) = \sum_{X_2} f(X_1, Y_1, Y_2, Y_3) \\
&f_3(Y_3) = \sum_{X_3} f(X_1, Y_1, Y_2, Y_3)
\end{align*}
\]

No hidden variables left. Join the remaining factors to get:

\[
P(X_3|Y_1, Y_2, Y_3) = \sum_{X_1} f(X_1, Y_1, Y_2, Y_3)
\]

Finally, solve \( P(X_1|\) given \( P(X_2|X_1, X_3, Y_1, Y_2, Y_3) \).

Variable Elimination Ordering

- For the query \( P(X_n|y_1, \ldots, y_n) \) work through the following two different orderings as done in previous slide \( X_1, \ldots, X_m \) and \( X_1, \ldots, X_{m-1}, Z \). What is the size of the maximum factor generated for each of the orderings?

- Answer: \( 2^n \) versus \( 2 \) (assuming binary)

- In general: the ordering can greatly affect efficiency.

VE: Computational and Space Complexity

- The computational and space complexity of variable elimination is determined by the largest factor.

- The elimination ordering can greatly affect the size of the largest factor.
  - E.g., previous slide’s example \( 2^n \) vs. \( 2 \)

- Does there always exist an ordering that only results in small factors?
  - No!

Worst Case Complexity?

- CSP:

- If we can answer \( P(z) \) equal to zero or not, we answered whether the 3-SAT problem has a solution.

- Hence inference in Bayes’ nets is NP-hard. No known efficient probabilistic inference in general.

Polytrees

- A polytree is a directed graph with no undirected cycles

- For poly-trees you can always find an ordering that is efficient
  - Try all!

- Cut-set conditioning for Bayes’ net inference
  - Choose set of variables such that if removed only a polytree remains
  - Exercise: Think about how the specifics would work out!

Bayes’ Nets

- Representation
- Conditional Independences
- Probabilistic Inference
  - Enumeration (exact, exponential complexity)
  - Variable elimination (exact, worst case exponential complexity, often better)
  - Inference is NP-complete
  - Sampling (approximate)
- Learning Bayes’ Nets from Data