Bayesian Networks – Inference (cont.)

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University
March 26th, 2006

Required Readings from Koller & Friedman:
Representation: 2.1, 2.2
Inference: 5.1, 6.1, 6.2, 6.7.1
Optional:
2.3, 5.2, 5.3, 6.3, 6.7.2
Marginalization

\[ P(F, S, N) = P(F) \cdot P(S|F) \cdot P(N|S) \]

\[ P(F = t, N = \epsilon) = \sum_s P(F = t, S = s, N = \epsilon) \]
\[ = P(F = t) \cdot P(S = t | F = t) \cdot P(N = \epsilon | S = \epsilon) \]
\[ + P(F = t) \cdot P(S = \epsilon | F = t) \cdot P(N = \epsilon | S = \epsilon) \]
Probabilistic inference example

\[ P(F = t, N = t) = \sum_{a, s, h} P(F = t, A = a, S = s, H = h, N = t) \]
\[ 8 = 2^3 \]

\[ P(F) P(A) P(S | FA) P(H | IS) P(N | S) \]

(4 vars., each with 3 vals.
\[ P(A = t, B = t) \Rightarrow \text{sum } 3^8 \]

Inference seems exponential in number of variables!
Understanding variable elimination – Order can make a HUGE difference

\[ P(F, N) = \sum_{a, s} P(F) \cdot P(a) \cdot P(s|Fa) \cdot P(N|s) \cdot P(s) \]

\[ g(F, a, N, s) \]

16 entries
Variable elimination algorithm

- Given a BN and a query $P(X|e) \propto P(X,e)$
- Instantiate evidence $e$
- Prune non-ancestors of $\{X,e\}$
- Choose an ordering on variables, e.g., $X_1, \ldots, X_n$
- For $i = 1$ to $n$, If $X_i \notin \{X,e\}$
  - Collect factors $f_1, \ldots, f_k$ that include $X_i$
  - Generate a new factor by eliminating $X_i$ from these factors

\[ g = \sum_{X_i} \prod_{j=1}^{k} f_j \]

- Variable $X_i$ has been eliminated!
- Normalize $P(X,e)$ to obtain $P(X|e)$
Complexity of variable elimination – (Poly)-tree graphs

Variable elimination order:
Start from “leaves” up – find topological order, eliminate variables in reverse order

Linear in number of variables!!! (versus exponential)
Complexity of variable elimination – Graphs with loops

Exponential in number of variables in largest factor generated
Complexity of variable elimination – Tree-width

Moralize graph:
Connect parents into a clique and remove edge directions

Complexity of VE elimination:
(“Only”) exponential in tree-width
Tree-width is maximum node cut +1
Example: Large tree-width with small number of parents

Compact representation $\Rightarrow$ Easy inference 😞
Choosing an elimination order

- Choosing best order is NP-complete
  - Reduction from MAX-Clique
- Many good heuristics (some with guarantees)
- Ultimately, can’t beat NP-hardness of inference
  - Even optimal order can lead to exponential variable elimination computation
- In practice
  - Variable elimination often very effective
  - Many (many many) approximate inference approaches available when variable elimination too expensive
Most likely explanation (MLE)

Query: $\arg\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n \mid c)$

Using Bayes rule:
$$\arg\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n \mid e) = \arg\max_{x_1,\ldots,x_n} \frac{P(x_1,\ldots,x_n, e)}{P(e)}$$

Normalization irrelevant:
$$\arg\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n \mid e) = \arg\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n, e)$$
Max-marginalization

Flu → Sinus → Nose=t
Example of variable elimination for MLE – Forward pass
Example of variable elimination for MLE – Backward pass
MLE Variable elimination algorithm – Forward pass

- Given a BN and a MLE query $\max_{x_1,\ldots,x_n} P(x_1,\ldots,x_n,e)$
- Instantiate evidence $e$
- Choose an ordering on variables, e.g., $X_1, \ldots, X_n$
- For $i = 1$ to $n$, if $X_i \notin \{e\}$
  - Collect factors $f_1,\ldots,f_k$ that include $X_i$
  - Generate a new factor by eliminating $X_i$ from these factors
    $$g = \max_{x_i} \prod_{j=1}^{k} f_j$$
  - Variable $X_i$ has been eliminated!
MLE Variable elimination algorithm – Backward pass

- \( \{x_1^*, \ldots, x_n^*\} \) will store maximizing assignment

For \( i = n \) to 1, if \( X_i \notin \{e\} \)

- Take factors \( f_1, \ldots, f_k \) used when \( X_i \) was eliminated
- Instantiate \( f_1, \ldots, f_k \), with \( \{x_{i+1}^*, \ldots, x_n^*\} \)
  - Now each \( f_j \) depends only on \( X_i \)
- Generate maximizing assignment for \( X_i \):

\[
x_i^* \in \text{argmax}_{x_i} \prod_{j=1}^{k} f_j
\]
What you need to know

- Bayesian networks
  - A useful compact representation for large probability distributions

- Inference to compute
  - Probability of X given evidence e
  - Most likely explanation (MLE) given evidence e
  - Inference is NP-hard

- Variable elimination algorithm
  - Efficient algorithm (“only” exponential in tree-width, not number of variables)
  - Elimination order is important!
  - Approximate inference necessary when tree-width too large
    - Not covered this semester
  - Only difference between probabilistic inference and MLE is “sum” versus “max”
HMMs

Machine Learning – 10701/15781
Carlos Guestrin
Carnegie Mellon University
March 26th, 2005

Classic HMM tutorial – see class website:
Adventures of our BN hero

- Compact representation for probability distributions
- Fast inference
- Fast learning

But... Who are the most popular kids?

1. Naïve Bayes
2 and 3. Hidden Markov models (HMMs) Kalman Filters
Handwriting recognition

Character recognition, e.g., kernel SVMs
Example of a hidden Markov model (HMM)
Understanding the HMM Semantics

\[ X_1 = \{a, \ldots, z\} \rightarrow X_2 = \{a, \ldots, z\} \rightarrow X_3 = \{a, \ldots, z\} \rightarrow X_4 = \{a, \ldots, z\} \rightarrow X_5 = \{a, \ldots, z\} \]
HMMs semantics: Details

Just 3 distributions:

\[ P(X_1) \]

\[ P(X_i \mid X_{i-1}) \]

\[ P(O_i \mid X_i) \]
HMMs semantics: Joint distribution

\[ P(X_1, \ldots, X_n \mid o_1, \ldots, o_n) = P(X_{1:n} \mid o_{1:n}) \]
\[ \propto P(X_1)P(o_1 \mid X_1) \prod_{i=2}^{n} P(X_i \mid X_{i-1})P(o_i \mid X_i) \]
Learning HMMs from fully observable data is easy

Learn 3 distributions:

\[ P(X_1) \]

\[ P(O_i \mid X_i) \]

\[ P(X_i \mid X_{i-1}) \]
Possible inference tasks in an HMM

Marginal probability of a hidden variable:

Viterbi decoding – most likely trajectory for hidden vars:
Using variable elimination to compute $P(X_i | o_{1:n})$

Variable elimination order?

Example:
What if I want to compute $P(X_i | o_{1:n})$ for each $i$?

Variable elimination for each $i$?

Variable elimination for each $i$, what’s the complexity?
Reusing computation

$X_1 = \{a, \ldots, z\} \rightarrow X_2 = \{a, \ldots, z\} \rightarrow X_3 = \{a, \ldots, z\} \rightarrow X_4 = \{a, \ldots, z\} \rightarrow X_5 = \{a, \ldots, z\}$

Compute:

$P(X_i \mid o_1..n)$
The forwards-backwards algorithm

\[ P(X_i \mid o_{1..n}) \]

- **Initialization:** \( \alpha_1(X_1) = P(X_1)P(o_1 \mid X_1) \)
- For \( i = 2 \) to \( n \)
  - Generate a forwards factor by eliminating \( X_{i-1} \)
  
  \[
  \alpha_i(X_i) = \sum_{x_{i-1}} P(o_i \mid X_i)P(X_i \mid X_{i-1} = x_{i-1})\alpha_{i-1}(x_{i-1})
  \]

- **Initialization:** \( \beta_n(X_n) = 1 \)
- For \( i = n-1 \) to \( 1 \)
  - Generate a backwards factor by eliminating \( X_{i+1} \)
  
  \[
  \beta_i(X_i) = \sum_{x_{i+1}} P(o_{i+1} \mid x_{i+1})P(x_{i+1} \mid X_i)\beta_{i+1}(x_{i+1})
  \]

- \( \forall \ i, \) probability is: \( P(X_i \mid o_{1..n}) = \alpha_i(X_i)\beta_i(X_i) \)
Most likely explanation

Compute:

Variable elimination order?

Example:
The Viterbi algorithm

- **Initialization:** \( \alpha_1(X_1) = P(X_1)P(o_1 \mid X_1) \)
- **For** \( i = 2 \) to \( n \)
  - Generate a forwards factor by eliminating \( X_{i-1} \)
  \[
  \alpha_i(X_i) = \max_{x_{i-1}} P(o_i \mid X_i)P(X_i \mid X_{i-1} = x_{i-1})\alpha_{i-1}(x_{i-1})
  \]
- **Computing best explanation:** \( x_n^* = \arg\max_{x_n} \alpha_n(x_n) \)
- **For** \( i = n-1 \) to \( 1 \)
  - Use \( \arg\max \) to get explanation:
  \[
  x_i^* = \arg\max_{x_i} P(x_{i+1}^* \mid x_i)\alpha_i(x_i)
  \]
What you’ll implement 1: multiplication

$$\alpha_i(X_i) = \max_{x_{i-1}} P(o_i \mid X_i) P(X_i \mid X_{i-1} = x_{i-1}) \alpha_{i-1}(x_{i-1})$$
What you’ll implement 2: max & argmax

\[ \alpha_i(X_i) = \max_{x_{i-1}} P(o_i \mid X_i) P(X_i \mid X_{i-1} = x_{i-1}) \alpha_{i-1}(x_{i-1}) \]
Higher-order HMMs

Add dependencies further back in time → better representation, harder to learn
What you need to know

- Hidden Markov models (HMMs)
  - Very useful, very powerful!
  - Speech, OCR,…
  - Parameter sharing, only learn 3 distributions
  - Trick reduces inference from $O(n^2)$ to $O(n)$
  - Special case of BN