Classic HMM tutorial – see class website:
*L. R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," Proc. of the IEEE, Vol.77, No.2, pp.257--286, 1989.

HMMs (cont.)

Machine Learning – 10701/15781

Carlos Guestrin
Carnegie Mellon University

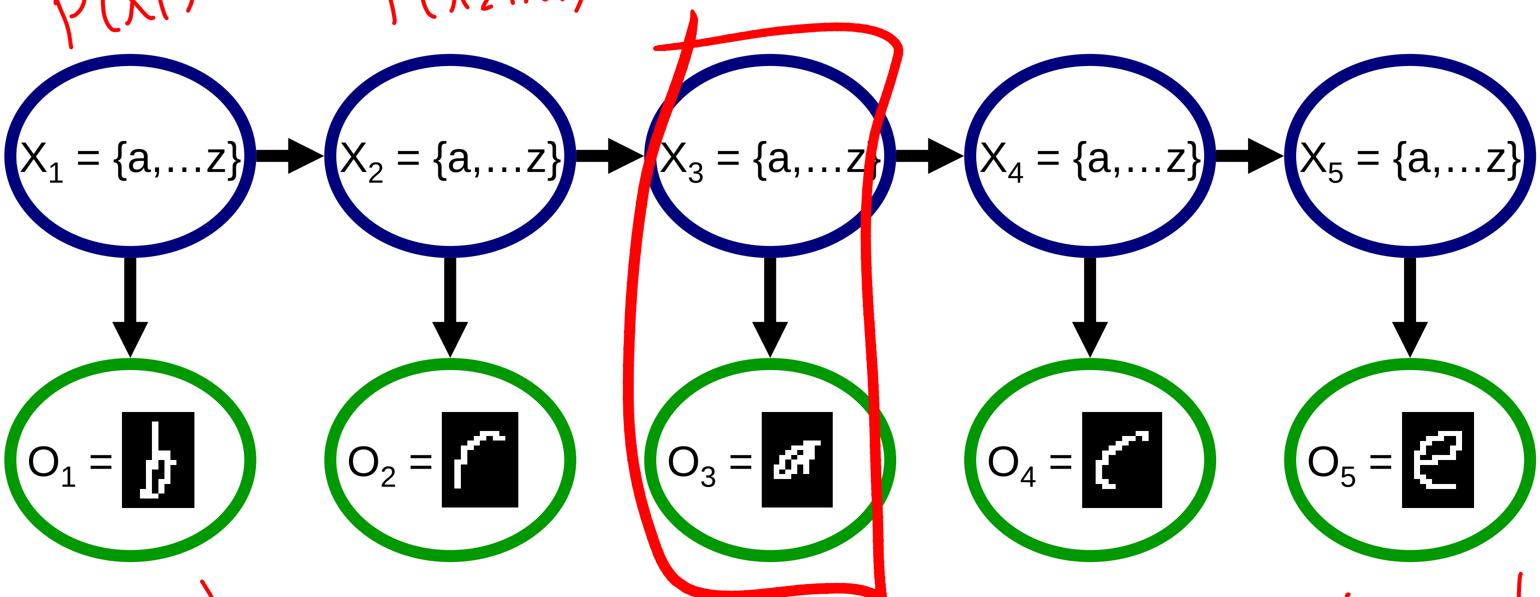
March 29th, 2006

Announcements

- This weeks recitation:
  - Go through several BNs topics, representation, inference, learning, in the context of an example $\rightarrow$ very useful for homework

Understanding the HMM Semantics

$P(X_1)$ $P(X_2|X_1)$



$P(O_1|X_1)$

one option is Naive Bayes

$P(O_1|X_1) = \prod_{j \in pixels} P(O_1^j|X_1)$

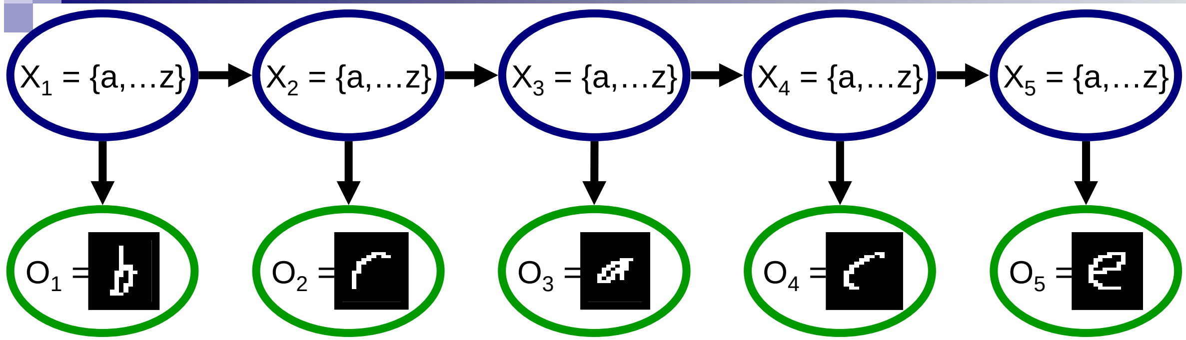
Markov assumption: the future is indep. of past given present

$X_{1:t-1} \perp X_{t+1:N} \mid X_t$

$O_t \perp$ everybody $\mid X_t$

basic!

HMMs semantics: Details



observing a after b has same prob. no matter where in word

a distribution

$P(X_3 | X_2) \sim P(X_2 | X_1)$

$\sim P(X_{10} | X_9)$

Just 3 distributions:

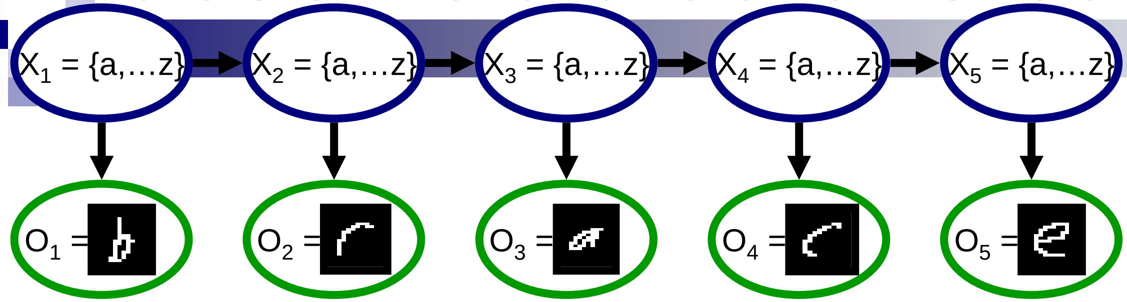
$$P(X_1)$$

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

$$P(O_i \mid X_i)$$

prob. image given letter is same for all positions in word.

Learning HMMs from fully observable data is easy



Learn 3 distributions:

$$P(X_1 = a) = \frac{\text{Count}(\#\text{ first letter was a})}{N = \text{dataset size}}$$

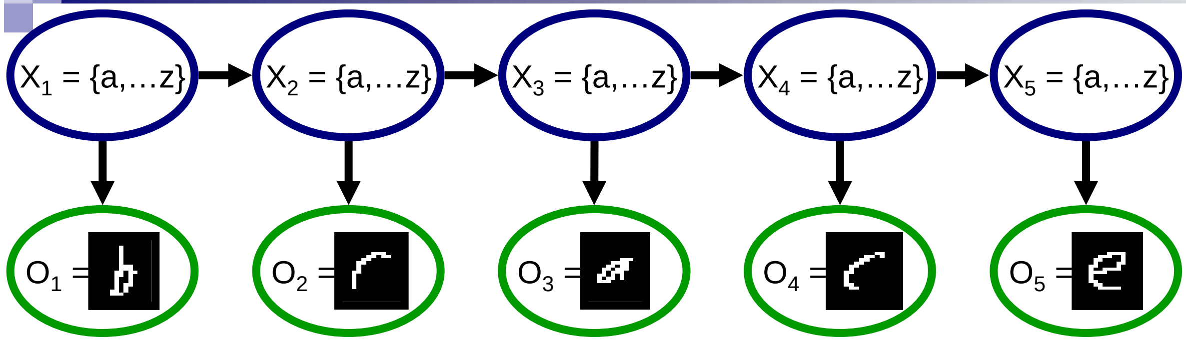
select training data where the letter was a

$$P(O_i = \text{pixel 17 is white} \mid X_i = a) = \frac{\text{Count}(\text{pixel 17 was white}, X_i = a)}{\text{Count}(X_i = a)}$$

any position in word

$$P(X_i = a \mid X_{i-1} = b) = \frac{\text{Count}(\text{a appears after b})}{\text{Count}(\#\text{ of b's that are not at the end of the word})}$$

Possible inference tasks in an HMM



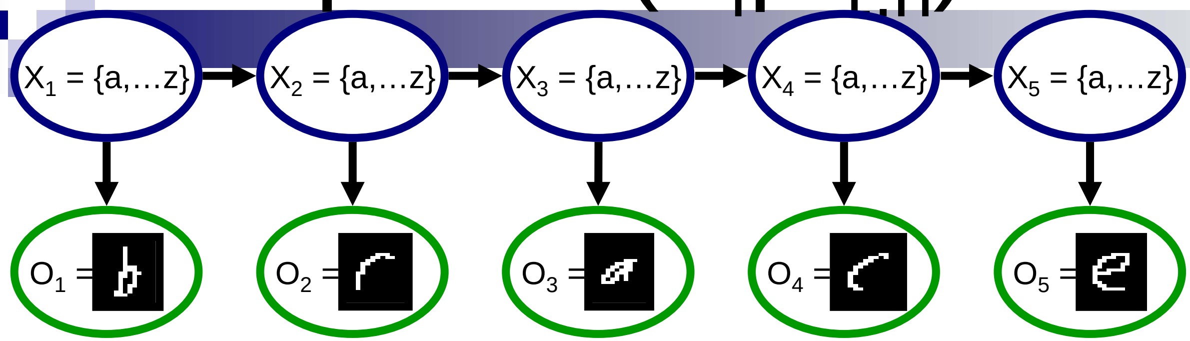
Marginal probability of a hidden variable:

$\rightarrow \quad P(X_i \mid O_1 = \boxed{b}, O_2 = \boxed{r}, O_3 = \boxed{a}, \ldots)$

Viterbi decoding – most likely trajectory for hidden vars:

$$\underset{x_1 x_2 x_3 x_4 x_5}{\operatorname{argmax}} \; P(x_1, x_2, x_3, x_4, x_5 \mid O_{1:5})$$

Using variable elimination to compute $P(X_i|o_{1:n})$



Compute:

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

Variable elimination order?

1, 2, 3, 4 .... n

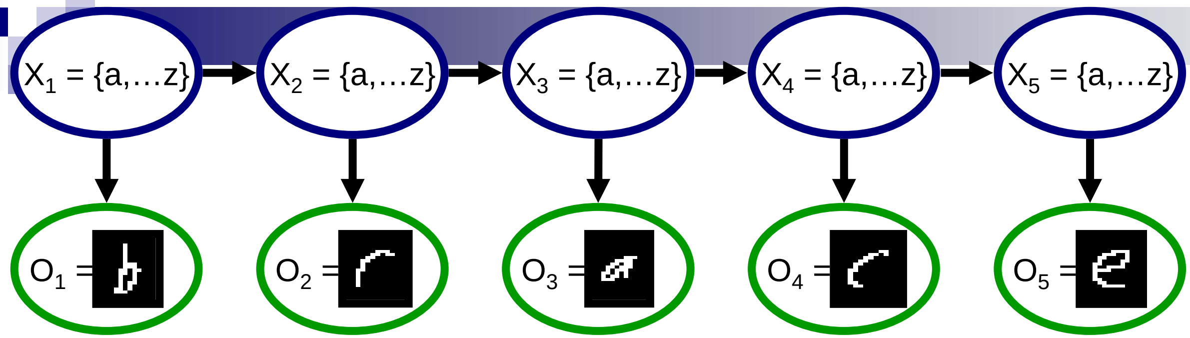
Example:

$$P(X_3 \mid O) = \sum_{X_1 X_2 X_4 X_5} P(x_1, x_2, X_3, x_4, x_5 \mid O)$$

$$\propto \sum_{x_1 x_2 x_4 x_5} P(x_1)\, P(O_1|X_1)\, P(X_2|X_1)\, P(O_2|X_2)\, P(X_3|X_2) \ldots$$

$$= \sum_{x_2 x_4 x_5} P(O_2|X_2)\, P(x_3|x_2) \ldots \underbrace{\sum_{x_1} P(x_1)\, P(o_1|X_1)\, P(X_2|X_1)}_{g_1(X_2, O_1)}$$

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



Compute:

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

Variable elimination for each i?

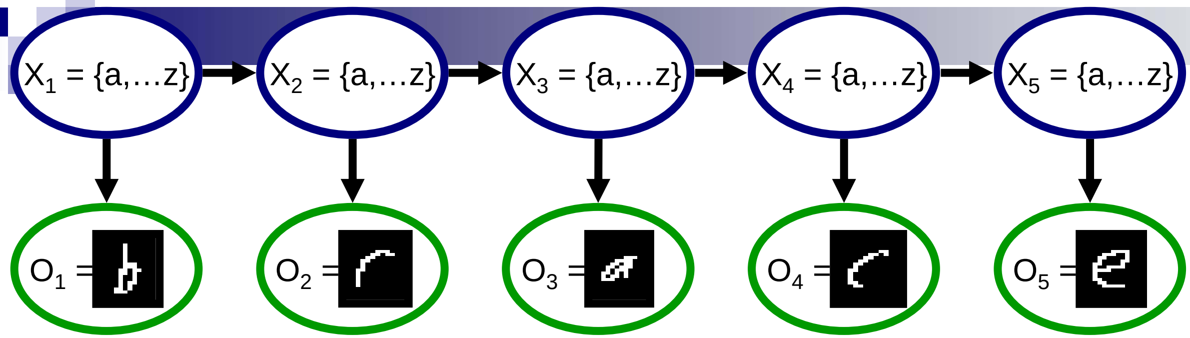
$P(X_1|O)$, $P(X_2|O)$, $P(X_3|O)$ ...

Variable elimination for each i, what's the complexity?

n letters $\quad P(X_i|O) \rightarrow O(n)$

$O(n^2)$ $\quad$ [can solve in $O(n)$

Reusing computation



Compute:

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

$P(X_5 | O) \rightarrow$ elimination order 1, 2, 3, 4

$g_1(X_2, O_1)$, $g_2(X_3, O_{1:2})$, $g_3(X_4, O_{1:3})$,

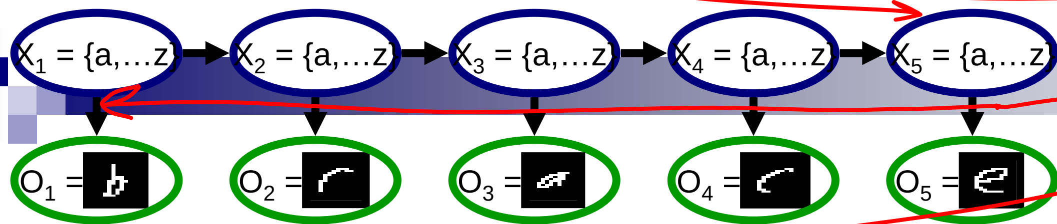
$g_4(X_5, O_{1:4})$

Same g's useful for $P(X_4|O)$, order 1, 2, 3, 5

eliminate 5 $\rightarrow$ use: $\underbrace{\sum_{X_5} P(O_5|X_5) \cdot P(X_5|X_4)}_{g_5(X_4, O_5)}$

computing $P(X_4|O)$: use g_1, g_2, g_3, g_5

α's The forwards-backwards algorithm



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

drop O_1 in notation

- 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}

sum out previous var prob obs transition prob

$$\alpha_i(X_i) = \sum_{x_{i-1}} P(o_i \mid X_i)P(X_i \mid X_{i-1} = x_{i-1})\alpha_{i-1}(x_{i-1})$$

$\alpha_n(X_n)$ normalized $= P(X_n \mid O_{1:n})$

$\beta_1(X_1)\alpha_1(X_1)$ normalized $= P(X_1 \mid O_{1:n})$

$\alpha_5(a)$
$\alpha_5(b)$
$\vdots$
$\alpha_5(z)$

- Initialization: $\beta_n(X_n) = 1$
- For i = n-1 to 1
  - Generate a backwards factor by eliminating X_{i+1}

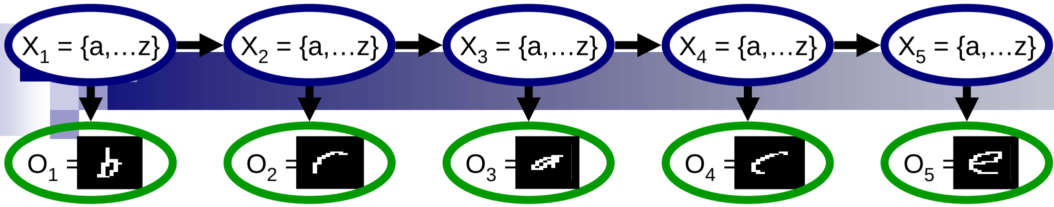
$\forall x'_i$

$$\beta_i(\overset{=x_i}{X_i}) = \sum_{x_{i+1}} P(o_{i+1} \mid x_{i+1})P(x_{i+1} \mid \overset{=x_i}{X_i})\beta_{i+1}(x_{i+1})$$

- $\forall$ i, probability is: $\boxed{P(X_i \mid o_{1..n}) \propto \alpha_i(X_i)\beta_i(X_i)}$

Most likely explanation

not equal to $\arg\max_{x_i} P(x_i \mid O_{1:5})$ $\forall i$



Compute: $\arg\max_{x_1 \ldots x_5} P(x_1, \ldots x_5 \mid O_{1:5})$

Variable elimination order?

1, 2, 3, 4, 5

Example:

$P(x_1, \ldots, x_5 \mid O_{1:5})$

$\max_{x_1 x_2 x_3 x_4, x_5}$

$= \max_{x_2, \ldots x_5} P(x_3 \mid x_2)\, P(O_2 \mid x_2)\, P(O_3 \mid x_3)\, P(x_4 \mid x_3) \ldots +$

$\underbrace{\max_{x_1} P(x_1) P(O_1 \mid x_1) P(x_2 \mid x_1)}_{\alpha_1(x_2)}$

$x_5^* = \arg\max_{x_5} \alpha_5(x_5)$

; backwards

$x_4^* = \arg\max_{x_4} \alpha_4(x_4)$.

$P(x_5^* \mid x_4)$

$P(O_5 \mid x_i^*)$

doesn't depend on X_4

A, B binary

$P(A,B) =$

B \ A	t	f
t	0.3	0
f	0.3	0.4

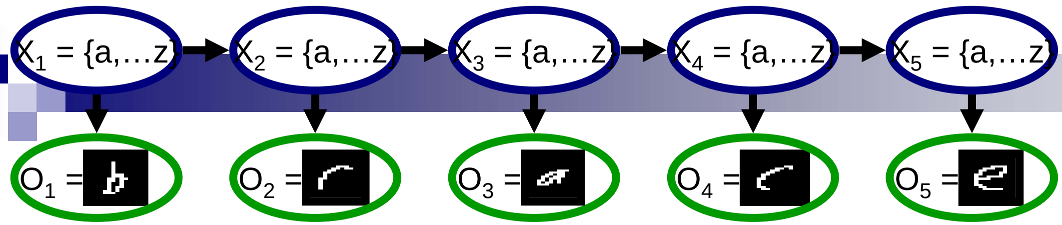
$\arg\max_{ab} P(a,b) = \begin{cases} a = f \\ b = f \end{cases}$

$P(a=t) = 0.6$

$P(a=f) = 0.4$

$\arg\max_{a} P(a) = (a = t)$

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}

$\forall x_i$

$$\alpha_i(\overset{=x_i}{X_i}) = \max_{x_{i-1}} \underbrace{P(o_i \mid \overset{=x_i}{X_i})}_{\text{obs. prob.}} \underbrace{P(\overset{=x_i}{X_i} \mid X_{i-1} = x_{i-1})}_{\text{transition}} \underbrace{\alpha_{i-1}(x_{i-1})}_{\text{message}}$$

- Computing best explanation: $x_n^* = \operatorname{argmax}_{x_n} \alpha_n(x_n)$
- For i = n-1 to 1
  - Use argmax to get explanation:

best for next letter

$$x_i^* = \operatorname{argmax}_{x_i} P(x_{i+1}^* \mid x_i)\alpha_i(x_i)$$

What you'll implement 1: multiplication

g_2

g_1

$$\alpha_i(X_i) = \max_{x_{i-1}} P(o_i \mid X_i) P(X_i \mid X_{i-1} = x_{i-1}) \alpha_{i-1}(x_{i-1})$$

$f_1(X_i) \cdot f_2(X_{i-1}, X_i)$ (factors)

def. new factor g, domain union $dom(f_1) \cup dom(f_2)$

$\forall x_i, x_{i-1}$

$g(X_{i-1} = x_{i-1}, X_i = x_i) = f_1(X_i = x_i) \cdot f_2(X_{i-1} = x_{i-1}, X_i = x_i)$

What you'll implement 2: max & argmax

$$\alpha_i(X_i) = \max_{x_{i-1}} \underbrace{P(o_i \mid X_i) P(X_i \mid X_{i-1} = x_{i-1}) \alpha_{i-1}(x_{i-1})}_{g(X_{i-1},\, X_i)}$$

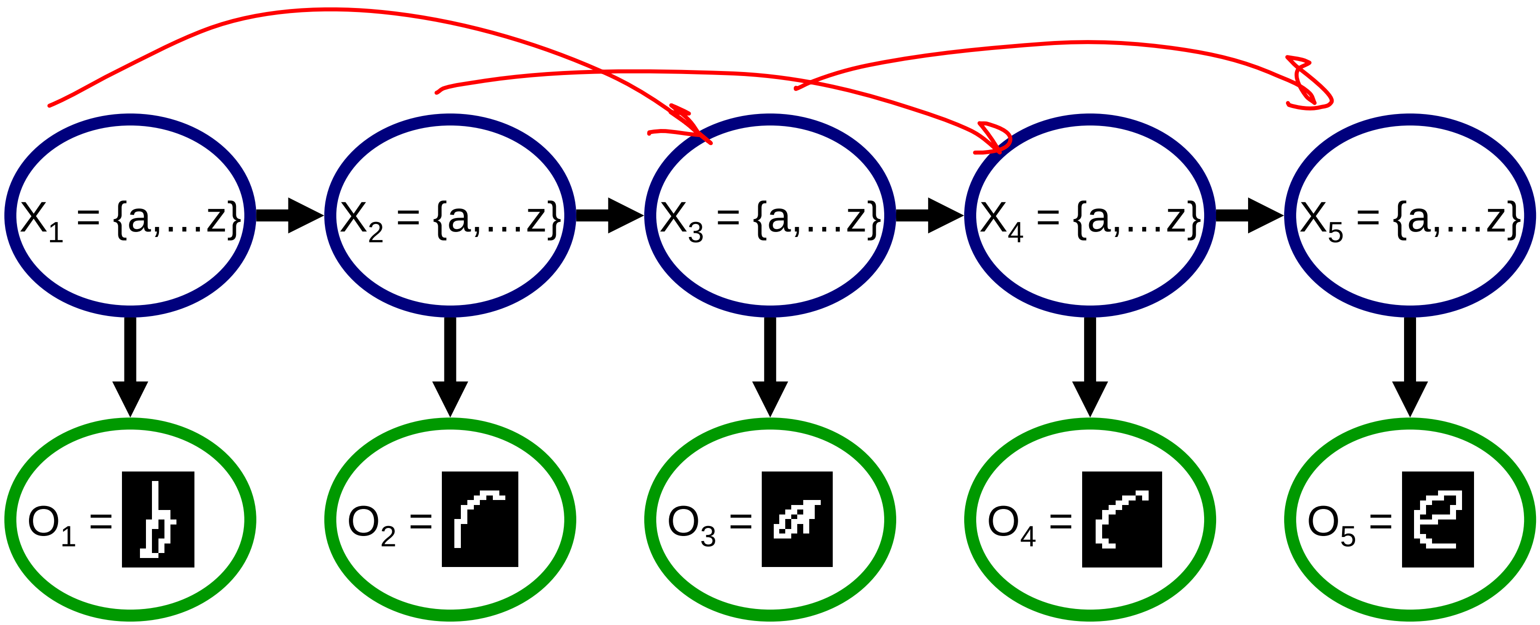
$$\underbrace{\max_{x_{i-1}} g(X_{i-1}, X_i)}_{\text{get function } h,\ \text{domain } \mathrm{dom}(g) - \{X_{i-1}\}}$$

$\forall x_i$

$$h(X_i = x_i) = \max_{x_{i-1}} g(X_{i-1} = x_{i-1}, X_i = x_i)$$

Higher-order HMMs

2nd order → depend on last 2 timesteps



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

Koller & Friedman Chapters (handed out):

- **Chapter 11 (short)**
- **Chapter 12: 12.1, 12.2, 12.3 (covered in the beginning of semester)**
  - **12.4 (Learning parameters for BNs)**
- **Chapter 13: 13.1, 13.3.1, 13.4.1, 13.4.3 (basic structure learning)**

Learning BN tutorial (class website):

- **ftp://ftp.research.microsoft.com/pub/tr/tr-95-06.pdf**

TAN paper (class website):

- **http://www.cs.huji.ac.il/~nir/Abstracts/FrGG1.html**

Bayesian Networks – (Structure) Learning

Machine Learning – 10701/15781

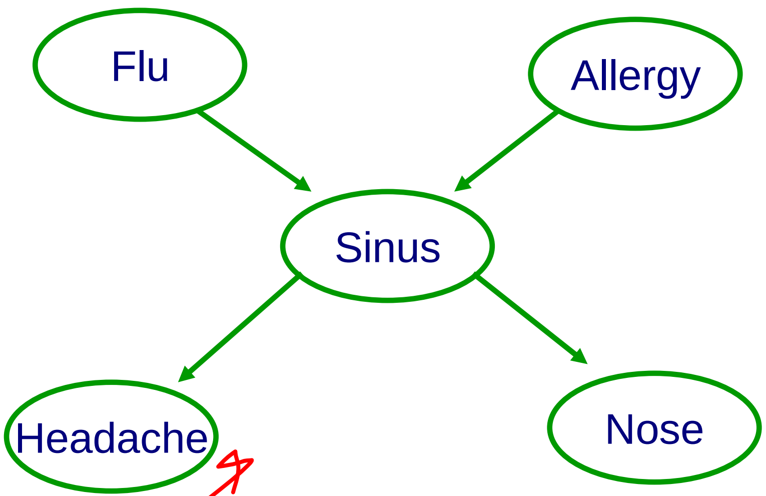
Carlos Guestrin

Carnegie Mellon University

March 29th, 2006

Review

- Bayesian Networks
  - Compact representation for probability distributions
  - Exponential reduction in number of parameters
- Fast probabilistic inference using variable elimination
  - Compute P(X|e)
  - Time exponential in tree-width, not number of variables
- Today
  - Learn BN structure



Learning Bayes nets

	Known structure	Unknown structure
Fully observable data	easy!! :)	NP-hard (not always) we'll see next week
Missing data	hard, in 2 weeks	really hard, next semester

A, ~~B~~, C

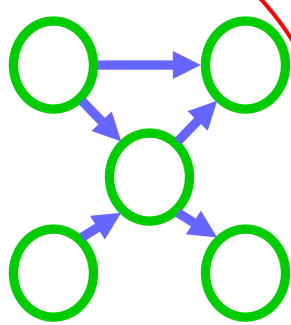
<A=a, B=?, C=c>

Data

$\mathbf{x}^{(1)}$

...

$\mathbf{x}^{(m)}$



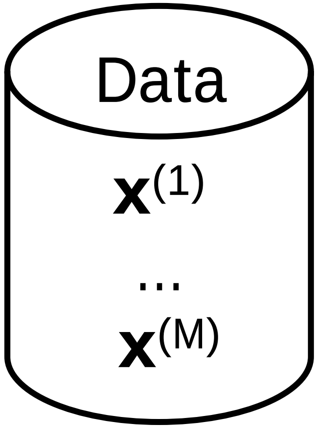
structure

+

CPTs – $P(X_i | \mathbf{Pa}_{X_i})$

parameters

Learning the CPTs

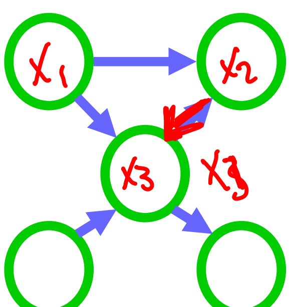


For each discrete variable X_i

$P(X_3 | X_1, X_2)$

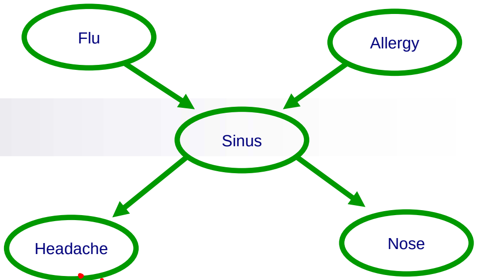
$$P(X_3 = x_3 | X_1 = x_1, X_2 = x_2) = \frac{\text{Count}(X_3 = x_3, X_1 = x_1, X_2 = x_2)}{\text{Count}(X_1 = x_1, X_2 = x_2)}$$

WHY??????????



MLE: $$P(X_i = x_i \mid X_j = x_j) = \frac{\text{Count}(X_i = x_i, X_j = x_j)}{\text{Count}(X_j = x_j)}$$

Information-theoretic interpretation of maximum likelihood



- Given structure, log likelihood of data:

$\arg\max_{\theta, G}$

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = \log \prod_j P(x^{(j)} \mid \theta, G)$$

$x^{(j)} = \langle F=t, S=f, A=f, H=t, N=f \rangle$

$$= \sum_j \log P(x^{(j)} \mid \theta, G)$$

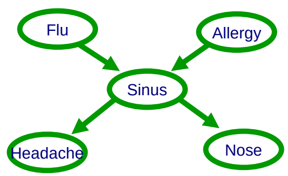
$$= \sum_j \log P(f^{(j)}) P(a^{(j)}) P(s^{(j)} \mid f^{(j)}, a^{(j)}) P(h^{(j)} \mid s^{(j)}) P(n^{(j)} \mid s^{(j)})$$

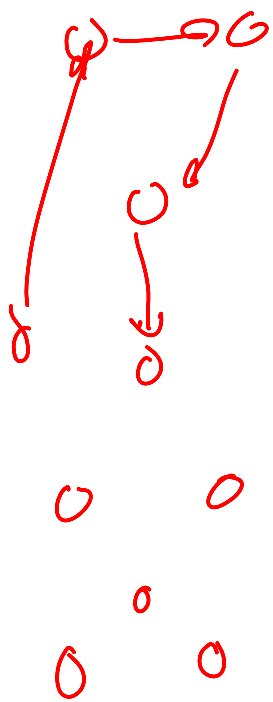
$$= \underbrace{\left[\sum_j \log P(f^{(j)})\right]}_{\text{learn } P(F)} + \underbrace{\left[\sum_j \log P(a^{(j)})\right]}_{\text{learn } P(A)} + \underbrace{\left[\sum_j \log P(s^{(j)} \mid f^{(j)}, a^{(j)})\right]}_{\text{learn } P(S \mid F, A)} + \dots$$

log likelihood decomposes with BN structure

Maximum likelihood (ML) for learning BN structure

Possible structures





Learn parameters using ML

Score structure

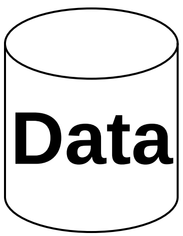
$\log P(D \mid \theta_G, G)$

-10000

wins

-12000

-50000

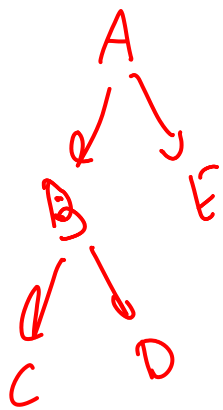


$\langle x_1^{(1)},\ldots,x_n^{(1)}\rangle$

…

$\langle x_1^{(M)},\ldots,x_n^{(M)}\rangle$

How many trees are there?



with n variables?

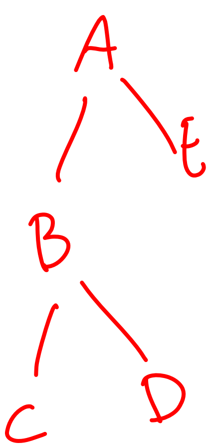
$$\boxed{\binom{n}{k} \ge \left(\frac{n}{k}\right)^k}$$

how many undirected trees

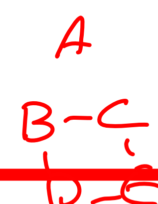
n vars $\rightarrow$ $\frac{n}{2}(n-1)$ possible edges

choosing $n-1$

$$\binom{\frac{n}{2}(n-1)}{n-1} \ge \left(\frac{\frac{n}{2}\,(n-1)}{n-1}\right)^{n-1} = \left(\frac{n}{2}\right)^{n-1}$$



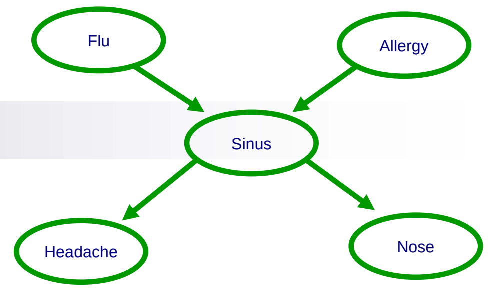
sorry... I counted



go back to d tree slide

Nonetheless – Efficient optimal algorithm finds best tree

Information-theoretic interpretation of maximum likelihood 2



- Given structure, log likelihood of data:

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = \sum_{i=1}^{n} \underbrace{\sum_{j=1}^{M} \log P(x_i^{(j)} \mid \mathbf{Pa}_{x_i,\mathcal{G}}^{(j)})}_{\text{learning for each CPT}}$$

$s=t,\ f=t,\ h=t$
$=50$ times

$$\sum_{j=1}^{M} \log P(x_i^{(j)} \mid Pa_{x_i,G}^{(j)}) = M \sum_{x_i, Pa_{x_i,G}} \frac{Count(x_i, Pa_{x_i,G})}{M} \cdot \log \hat{P}(x_i \mid Pa_{x_i,G})$$

$$\frac{Count(a,b)}{M} = \hat{P}(a,b)$$

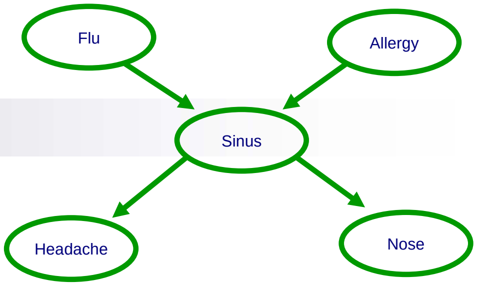
$$= M \sum_{x_i, Pa_{x_i,G}} \hat{P}(x_i, Pa_{x_i,G}) \log \hat{P}(x_i \mid Pa_{x_i,G})$$

$$= -M \hat{H}(X_i \mid Pa_{x_i,G})$$

Conditional entropy

$$H(A \mid B) = -\sum_{a,b} P(a,b) \log P(a \mid b)$$

Information-theoretic interpretation of maximum likelihood 3



- Given structure, log likelihood of data:

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = M \sum_{i=1}^{n} \sum_{x_i, \mathbf{Pa}_{x_i,\mathcal{G}}} \widehat{P}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) \log \widehat{P}(x_i \mid \mathbf{Pa}_{x_i,\mathcal{G}})$$

Mutual information $\rightarrow$ Independence tests

- Statistically difficult task!
- Intuitive approach: **Mutual information**

$$I(X_i, X_j) = \sum_{x_i, x_j} P(x_i, x_j) \log \frac{P(x_i, x_j)}{P(x_i)P(x_j)}$$

- Mutual information and independence:
  - X_i and X_j independent if and only if $I(X_i, X_j)=0$
- Conditional mutual information:

Decomposable score

- Log data likelihood

$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G}) = M \sum_i \hat{I}(X_i, \mathbf{Pa}_{X_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

Scoring a tree 1: equivalent trees

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

Scoring a tree 2: similar trees

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

Chow-Liu tree learning algorithm 1

- For each pair of variables X_i, X_j
  - Compute empirical distribution:

$$\widehat{P}(x_i, x_j) = \frac{\mathsf{Count}(x_i, x_j)}{M}$$

  - Compute mutual information:

$$\widehat{I}(X_i, X_j) = \sum_{x_i, x_j} \widehat{P}(x_i, x_j) \log \frac{\widehat{P}(x_i, x_j)}{\widehat{P}(x_i)\widehat{P}(x_j)}$$

- Define a graph
  - Nodes $X_1, \ldots, X_n$
  - Edge (i,j) gets weight $\widehat{I}(X_i, X_j)$

Chow-Liu tree learning algorithm 2

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

- Optimal tree BN
  - Compute maximum weight spanning tree
  - Directions in BN: pick any node as root, breadth-first-search defines directions

Can we extend Chow-Liu 1

- Tree augmented naïve Bayes (TAN) [Friedman et al. '97]
  - Naïve Bayes model overcounts, because correlation between features not considered
  - Same as Chow-Liu, but score edges with:

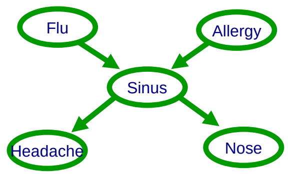
$$\hat{I}(X_i, X_j \mid C) = \sum_{c, x_i, x_j} \hat{P}(c, x_i, x_j) \log \frac{\hat{P}(x_i, x_j \mid c)}{\hat{P}(x_i \mid c)\hat{P}(x_j \mid c)}$$

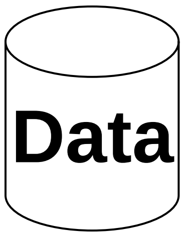
Can we extend Chow-Liu 2

- (Approximately learning) models with tree-width up to *k*
  - [Narasimhan & Bilmes ’04]
  - But, $O(n^{k+1})$…

Scoring general graphical models – Model selection problem

What's the best structure?





<x_1^{(1)},…,x_n^{(1)}>
…
<x_1^{(m)},…,x_n^{(m)}>

The more edges, the fewer independence assumptions, the higher the likelihood of the data, but will overfit…

Maximum likelihood overfits!

$$\log \hat{P}(\mathcal{D} \mid \theta, \mathcal{G}) = M \sum_i \hat{I}(x_i, \mathbf{Pa}_{x_i,\mathcal{G}}) - M \sum_i \hat{H}(X_i)$$

- Information never hurts:
- Adding a parent always increases score!!!

Bayesian score avoids overfitting

- Given a structure, distribution over parameters

$$\log P(D \mid \mathcal{G}) = \log \int_{\theta_{\mathcal{G}}} P(D \mid \mathcal{G}, \theta_{\mathcal{G}}) P(\theta_{\mathcal{G}} | \mathcal{G}) d\theta_{\mathcal{G}}$$

- Difficult integral: use Bayes information criterion (BIC) approximation (equivalent as M$\rightarrow \infty$)

$$\log P(D \mid \mathcal{G}) \approx \log P(D \mid \mathcal{G}, \theta_{\mathcal{G}}) - \frac{\text{NumberParams}(\mathcal{G})}{2} \log M + \mathcal{O}(1)$$

- Note: regularize with MDL score
- Best BN under BIC still NP-hard

How many graphs are there?

$$\sum_{k=1}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) = 2^n - 1$$

Structure learning for general graphs

- In a tree, a node only has one parent

- **Theorem**:
  - The problem of learning a BN structure with at most *d* parents is NP-hard for any (fixed) $d \geq 2$

- Most structure learning approaches use heuristics
  - Exploit score decomposition
  - (Quickly) Describe two heuristics that exploit decomposition in different ways

Learn BN structure using local search

Starting from Chow-Liu tree

Local search, possible moves:

- Add edge
- Delete edge
- Invert edge

Score using BIC

What you need to know about learning BNs

- Learning BNs
  - Maximum likelihood or MAP learns parameters
  - Decomposable score
  - Best tree (Chow-Liu)
  - Best TAN
  - Other BNs, usually local search with BIC score