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

Bayesian Networks – Inference (cont.)

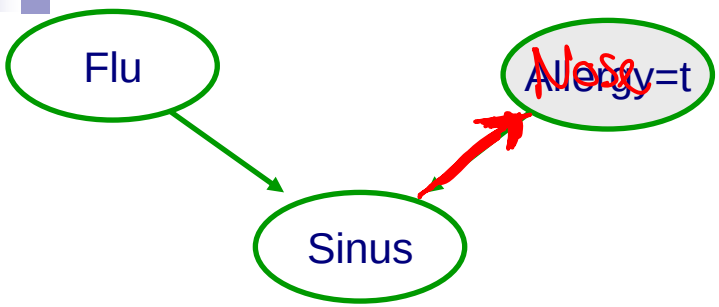
Machine Learning – 10701/15781

Carlos Guestrin

Carnegie Mellon University

March 26th, 2006

Marginalization



$P(F, S, N) = P(F) \cdot P(S|F) \cdot P(N|S)$

$P(F=t, N=t)$

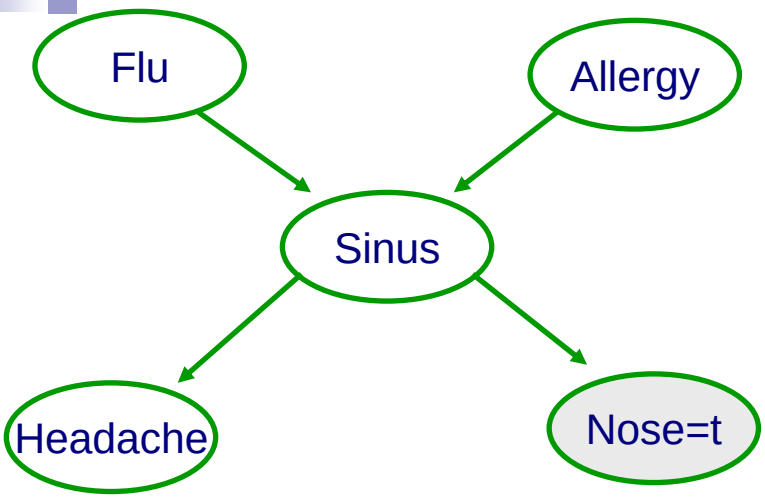
$= \sum_s P(F=t, S=s, N=t)$

$= P(F=t) \cdot P(S=t \mid F=t) \cdot P(N=t|S=t)$

$+$

$P(F=t) \cdot P(S=f \mid F=t) \cdot P(N=t|S=f)$

Probabilistic inference example



$P(F=t, N=t)$

$= \sum_a \sum_s \sum_h P(F=t, A=a, S=s, H=h, N=t)$

a, s, h: $8 = 2^3$

$P(F)\,P(A)\,P(S|FA)\,P(H|S)\,P(N|S)$

54 vars., each with 3 vals.

$P(A=t, B=t)$ => sum 3^{52}

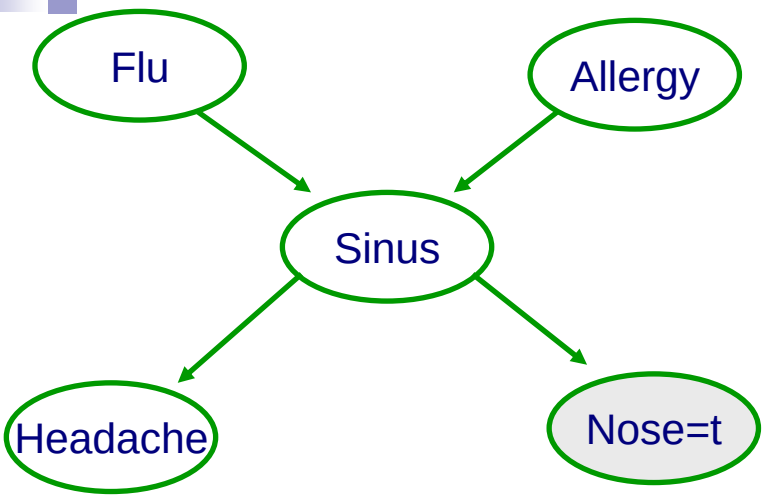
7 sums

4.8 multiplies

32

Inference seems exponential in number of variables!

Understanding variable elimination – Order can make a HUGE difference



$$P(F,N) = \sum_{ash} P(F)\cdot P(a)\, P(s|F,a)\, P(N|s)\, P(h|s)$$

$$= \sum_{ah} P(F)\, P(a) \sum_{s} P(s|Fa)\, P(N|s)\, P(h|s)$$

$g(F,a,N,h)$

16 entries

Variable elimination algorithm

initialize factors as all of CPTs

- Given a BN and a query $P(X|e) \propto P(X,e)$
- Instantiate evidence e — fix values, N = t
- Prune non-ancestors of {X,e} — **IMPORTANT!!!**
- 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 — CPTs, or g's
  - Generate a new factor by eliminating X_i from these factors

e.g.: $f_1: P(B|A)$, $f_2: P(C|B)$; $A \to B \to C$

$f_1(A,B)$, $f_2(B,C)$

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

$f(A,B,C) = f_1 \cdot f_2$

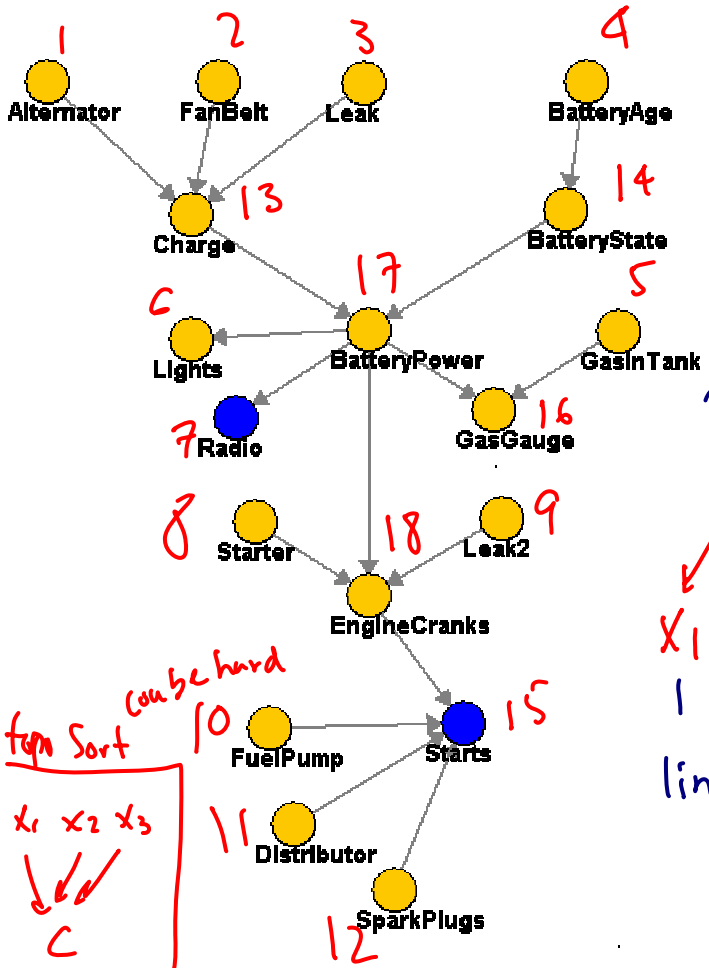
$f(A=t,B=t,C=f) = f_1(A=t,B=t) \cdot f_2(B=t,C=f)$

$\sum_S P(S|N,F) \cdot g(N,S)$

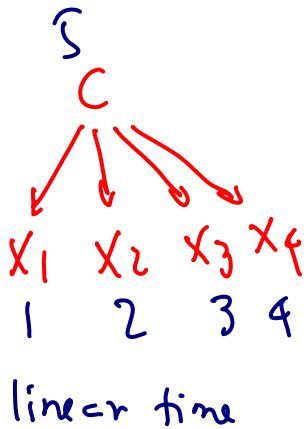
  - Variable X_i has been eliminated!
- Normalize P(X,e) to obtain P(X|e)

Complexity of variable elimination – (Poly)-tree graphs

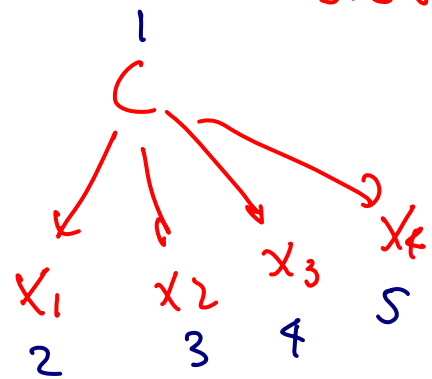
optimal order for poly-trees



Variable elimination order:
Start from "leaves" up – (ignore directions)
find ~~topological order~~, eliminate variables in ~~reverse~~ order



VE → linear in CPT sizes



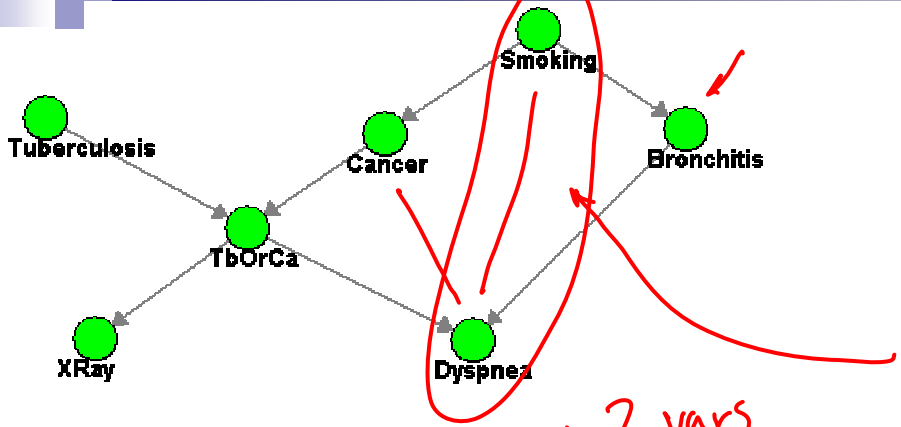
$$\sum_C P(C, X_{1:N}) = \sum_C P(C) \cdot \Pi_i P(X_i|C)$$

eliminate C: $g(X_1, X_2, X_3 \ldots X_n)$ exp.!

topo sort

Linear in number of variables!!! (versus exponential)

Complexity of variable elimination – Graphs with loops

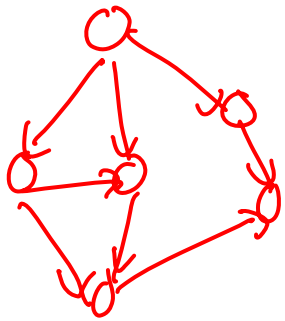


just starting from "leaves" not enough ...

larger factors

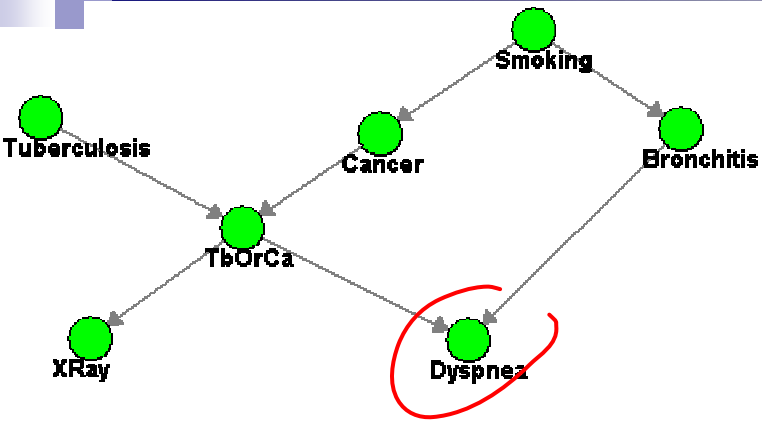
over 2 vars

$$g = \sum_{x_j} \prod_i f_i$$

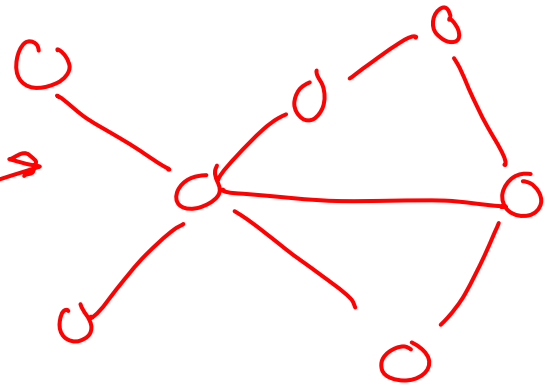


Exponential in number of variables in largest factor generated

Complexity of variable elimination – Tree-width



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



set of vars.

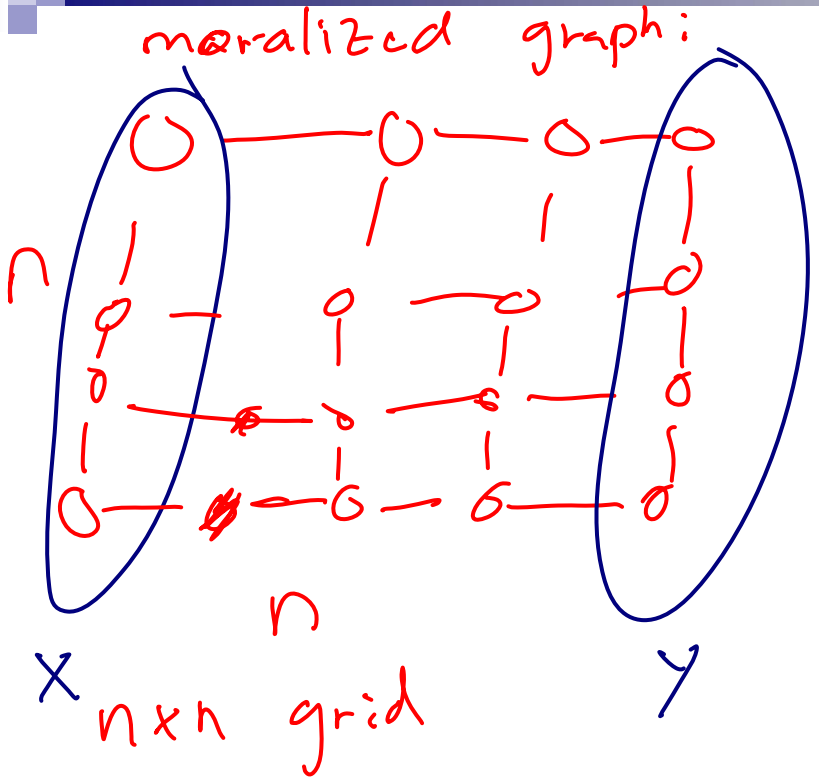
$\forall X, Y \subseteq V$, not directly connected

how many nodes do I have to remove to separate X, Y?

treewidth = 2 + 1 = 3

Complexity of VE elimination:
("Only") exponential in tree-width
Tree-width is maximum minimum node cut +1

Example: Large tree-width with small number of parents



tree-width?

n + 1

↳ def. of treewidth has a +1

even if vars have few parents, inference could be exponential in n

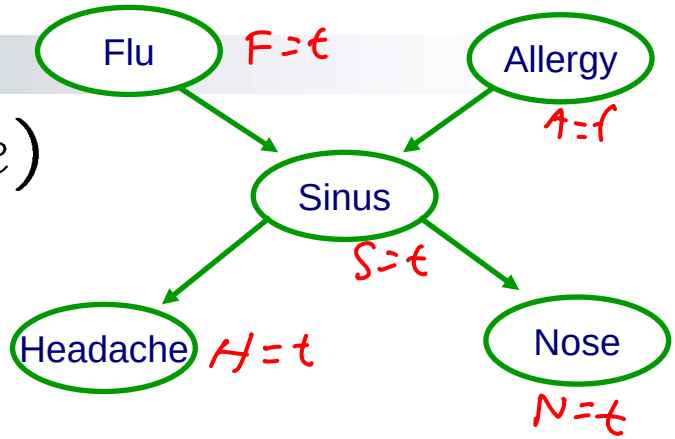
Compact representation ⇏ Easy inference ☹

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: $\underset{x_1,\ldots,x_n}{\operatorname{argmax}} P(x_1, \ldots, x_n \mid e)$



- Using Bayes rule:

$$\underset{x_1,\ldots,x_n}{\operatorname{argmax}} P(x_1, \ldots, x_n \mid e) = \underset{x_1,\ldots,x_n}{\operatorname{argmax}} \frac{P(x_1, \ldots, x_n, e)}{P(e)}$$

- Normalization irrelevant:

$$\underset{x_1,\ldots,x_n}{\operatorname{argmax}} P(x_1, \ldots, x_n \mid e) = \underset{x_1,\ldots,x_n}{\operatorname{argmax}} P(x_1, \ldots, x_n, e)$$

Max-marginalization

$\max_{fs} P(f) \cdot P(s|f) \cdot P(N=t|s) = \max_{f} P(f) \cdot \max_{s} P(s|f) P(N=t|s)$

$= \max_{f} P(f)\, g_1(f, N=t) = g_2(N=t)$

$g_1(f, N=t)$

$g_2(N=t)$: prob. of most likely f & s given N=t

$g_1(f, N=t)$: prob. of most likely s given f and N=t

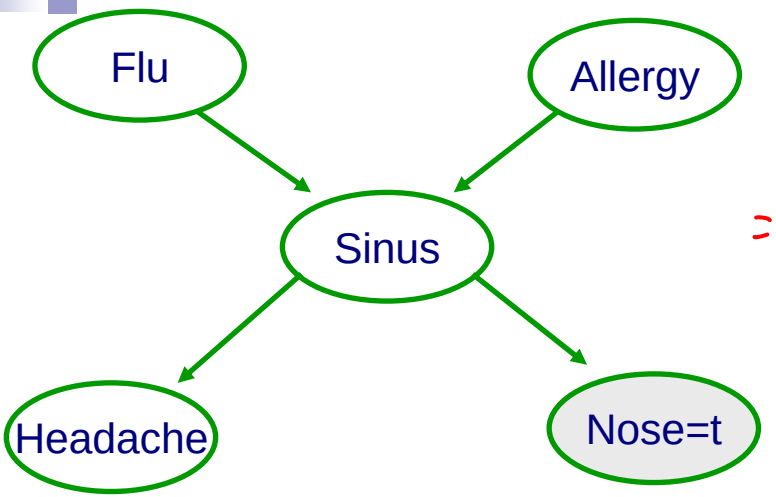
$\operatorname{argmax}_{fs} P(f, s, N=t)$

$\operatorname{argmax}_{fs}$?

$f^* = \operatorname{argmax}_{f} P(f) \cdot g_1(f, N=t)$

$s^* = \operatorname{argmax}_{s} P(s|f^*) P(N=t|s)$

Example of variable elimination for MLE – Forward pass



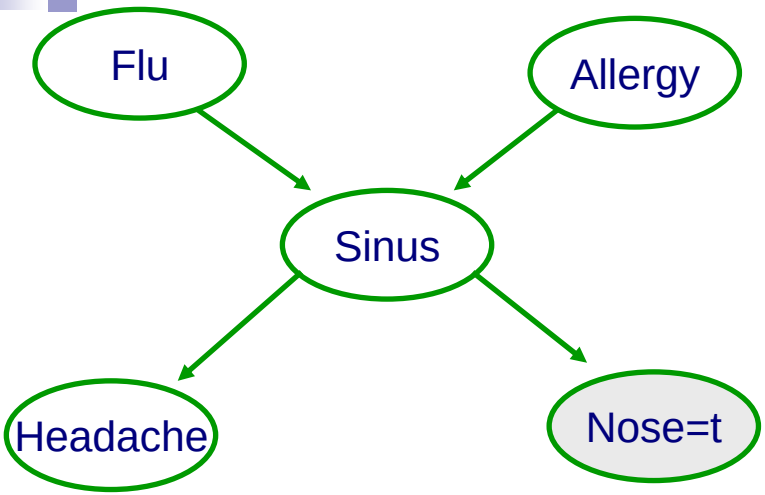
$$\max_{f,a,s,h} P(f)\cdot P(a)\cdot P(s|f,a)\cdot P(h|s)\cdot P(N=t|s)$$

$$= \max_{fas} P(f)\,P(a)\,P(s|fa)\,P(N=t|s)\underbrace{\max_{h} P(h|s)}_{g_1(s)}$$

$$= \max_{fa} P(f)\,P(a)\underbrace{\max_{s} P(s|fa)\,P(N=t|s)\,g_1(s)}_{g_2(f,a,N=t)}$$

$$= \max_{f} P(f)\underbrace{\max_{a} P(a)\,g_2(f,a,N=t)}_{g_3(f,N=t)}$$

Example of variable elimination for MLE – Backward pass



$$f^* = \arg\max_f P(f)\ g_3(f, N=t)$$

$$a^* = \arg\max_a P(a)\ g_2(f^*, a, N=t)$$

$$\vdots$$

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

only diff. is max v. $\sum$

$$g = \max_{x_i} \prod_{j=1}^{k} f_j$$

  - 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:

$$x_i^* \in \arg\max_{x_i} \prod_{j=1}^{k} f_j$$

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 to large
    - not covered this semester
  - Only difference between probabilistic inference and MLE is "sum" versus "max"

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

Machine Learning – 10701/15781

Carlos Guestrin

Carnegie Mellon University

March 26th, 2005

Adventures of our BN hero

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

- But… Who are the most popular kids?

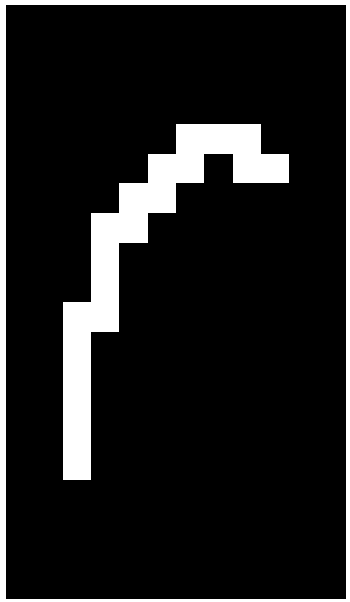
1. Naïve Bayes

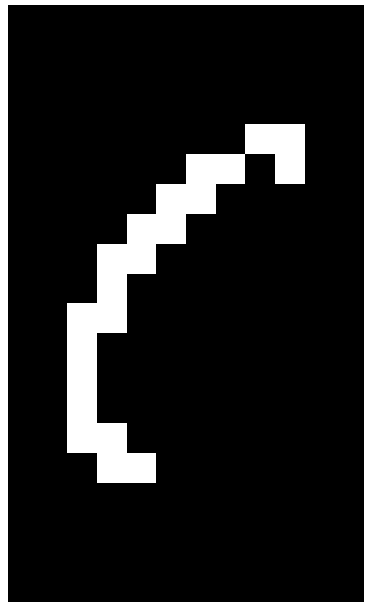
discrete vars.

2 and 3.
Hidden Markov models (HMMs)
Kalman Filters

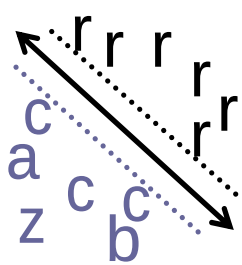
Gaussian vars.

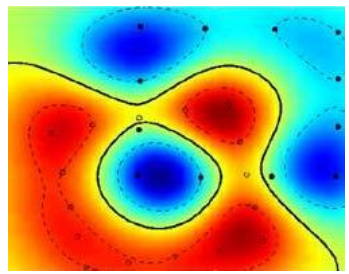
Handwriting recognition



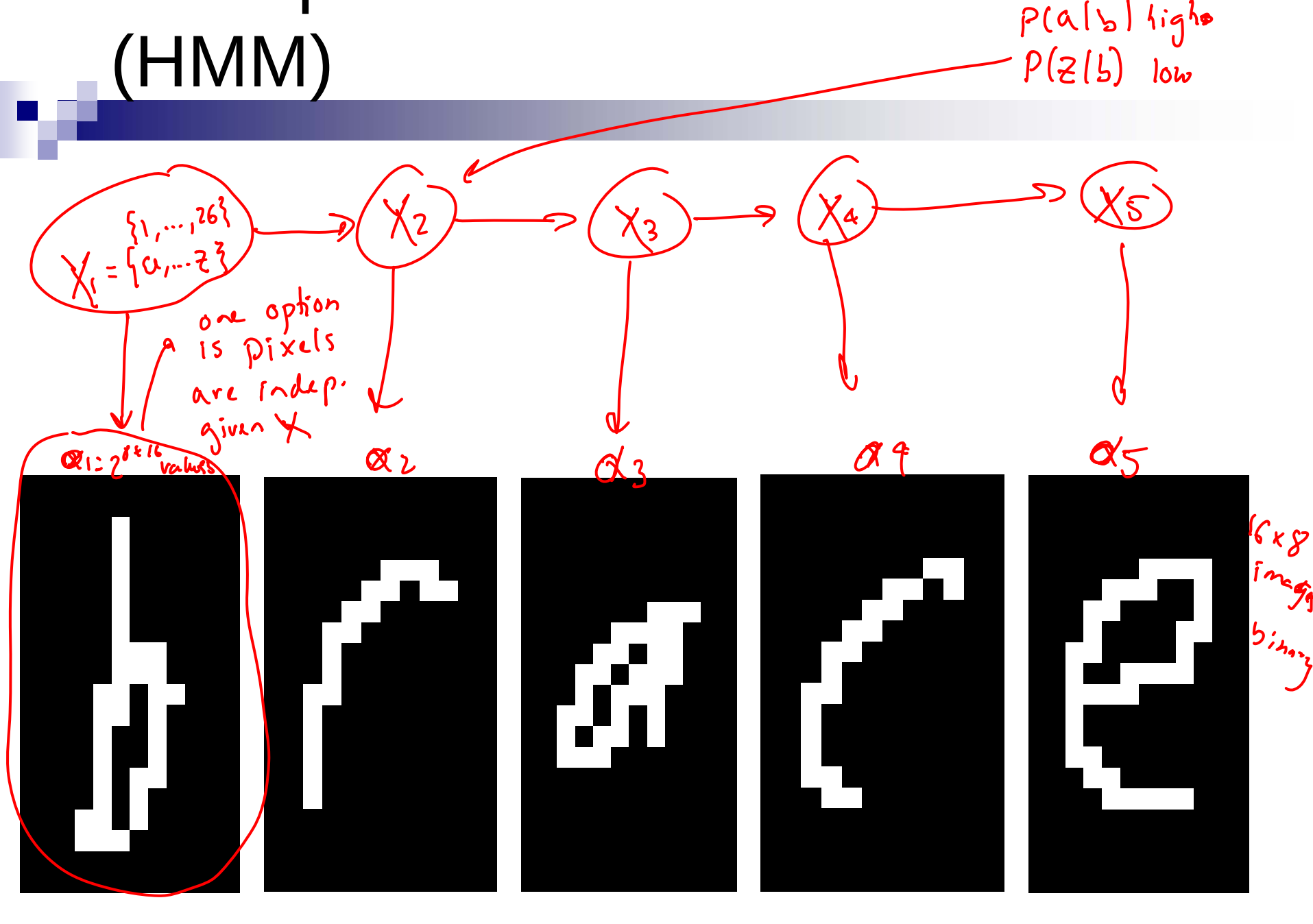


Character recognition, e.g., kernel SVMs

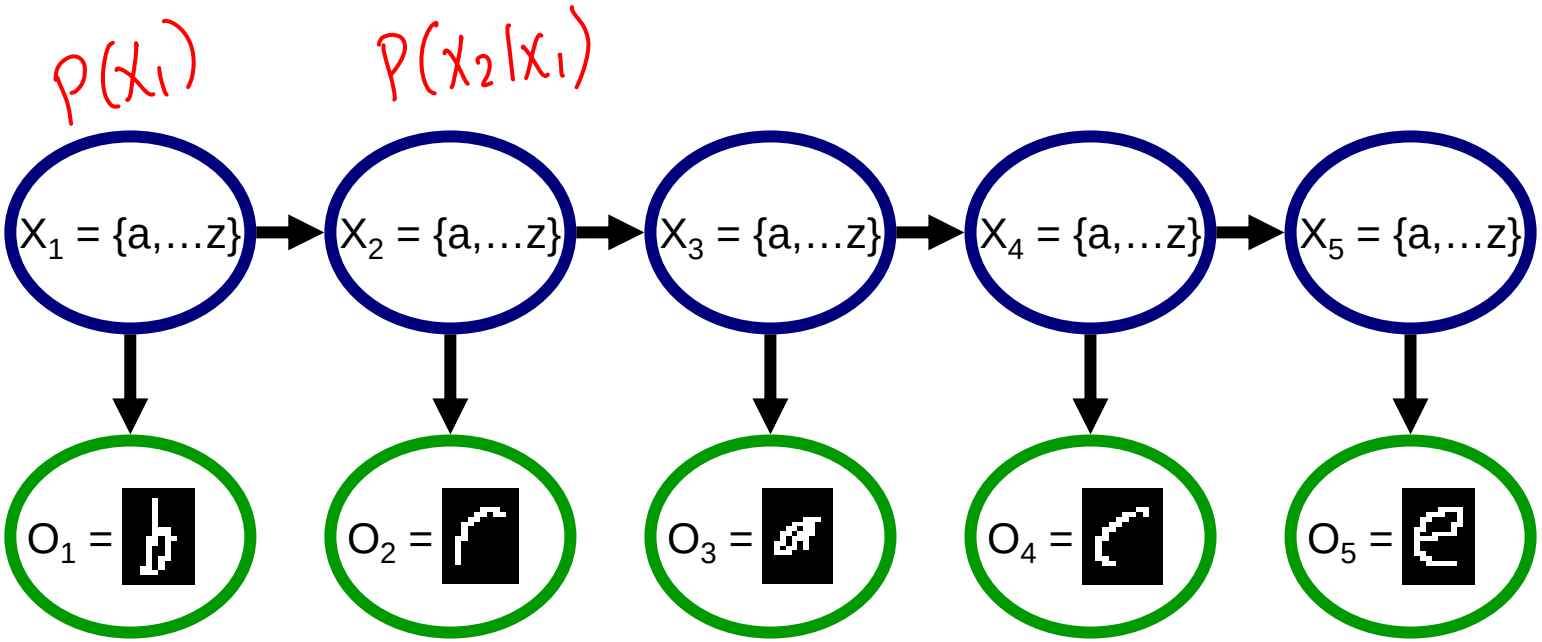




Example of a hidden Markov model (HMM)



Understanding the HMM Semantics



$P(X_1)$

$P(X_2|X_1)$

$P(O_1|X_1)$

one option is Naive Bayes

$P(O_1|X_1) = \prod_{\text{pixels } i} P(O_i|X_1)$

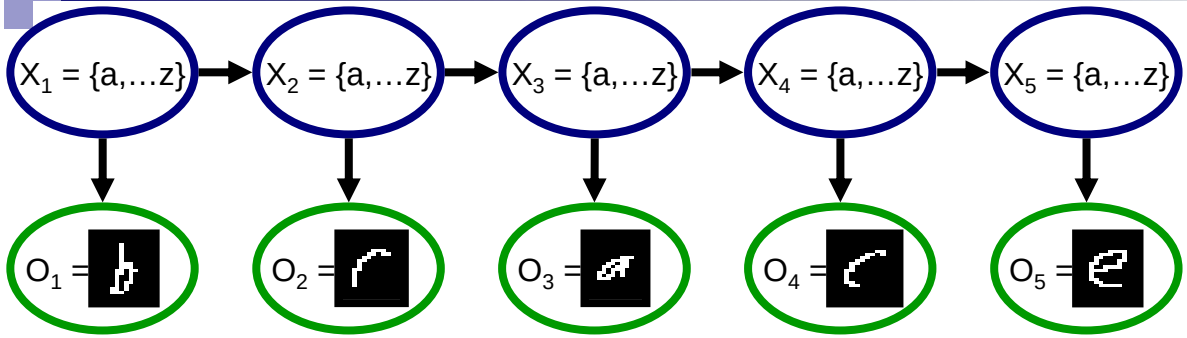
Markov assumption: the future is indep. of past given present

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

$O_t \perp$ everybody $\mid X_t$

basic!

HMMs semantics: Details



Just 3 distributions:

$$P(X_1)$$

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

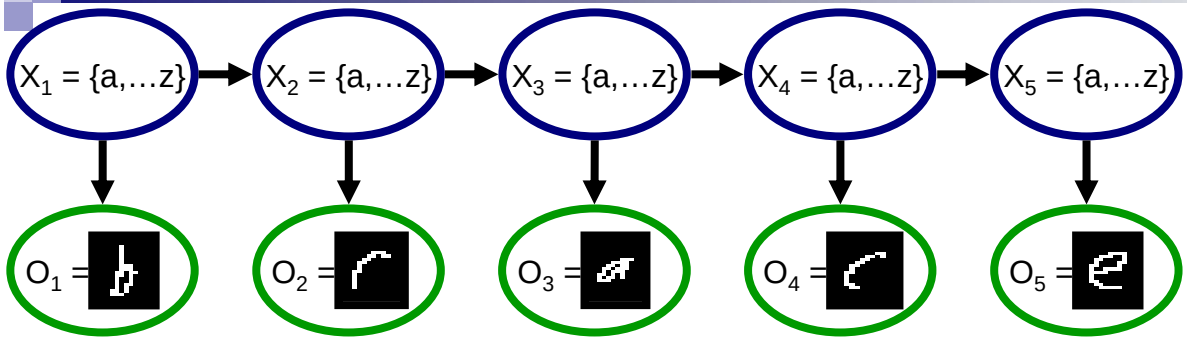
$$P(O_i \mid X_i)$$

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

a distribution $P(X_3 \mid X_2) \sim P(X_2 \mid X_1) \sim P(X_{10} \mid X_9)$

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

HMMs semantics: Joint distribution

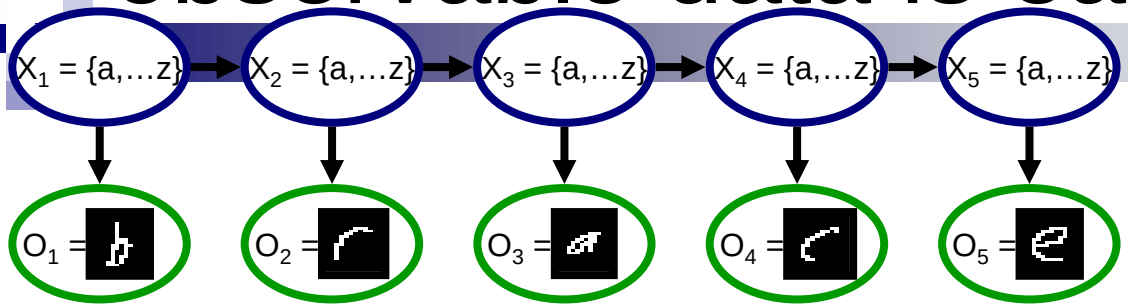


$$P(X_1)$$
$$P(X_i \mid X_{i-1})$$
$$P(O_i \mid X_i)$$

$P(X_1, X_2, X_3, X_4, X_5 \mid O_1, O_2, O_3, O_4, O_5) \propto P(X_1) \cdot P(O_1 = \boxed{5} \mid X_1)\, P(X_2 \mid X_1) \cdot P(O_2 = \boxed{C} \mid X_2)\, P(X_3 \mid X_2) \ldots$

$$P(X_1, \ldots, X_n \mid o_1, \ldots, o_n) = P(X_{1:n} \mid o_{1:n})$$
$$\propto P(X_1) P(o_1 \mid X_1) \prod_{i=2}^{n} P(X_i \mid X_{i-1}) P(o_i \mid X_i)$$

Learning HMMs from fully observable data is easy



Learn 3 distributions:

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

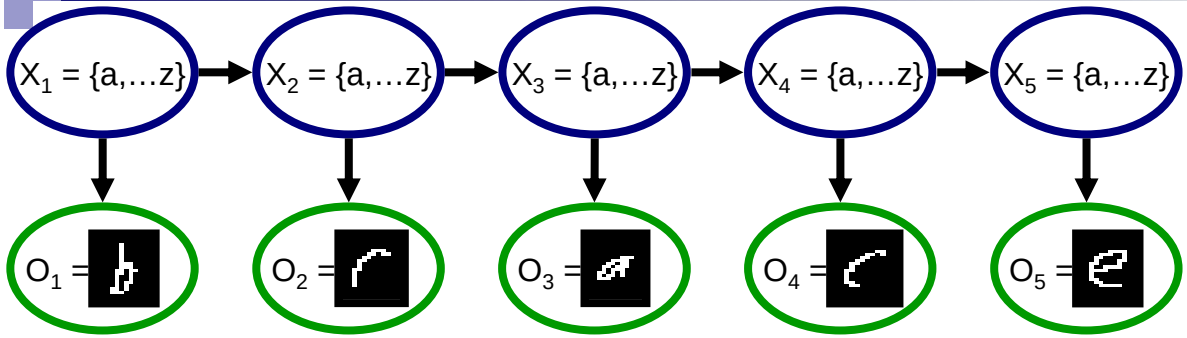
select training data where letter was a

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

Count any position in word

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

Possible inference tasks in an HMM



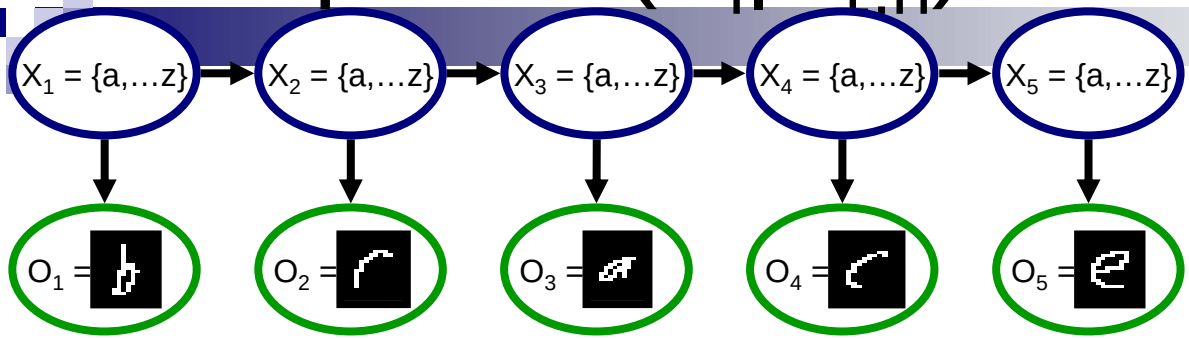
Marginal probability of a hidden variable:

$p(x_i \mid o_1 = \boxed{b}, o_2 = \boxed{r}, o_3 = \boxed{a}, \ldots)$

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 $P(X_i|o_{1:n})$



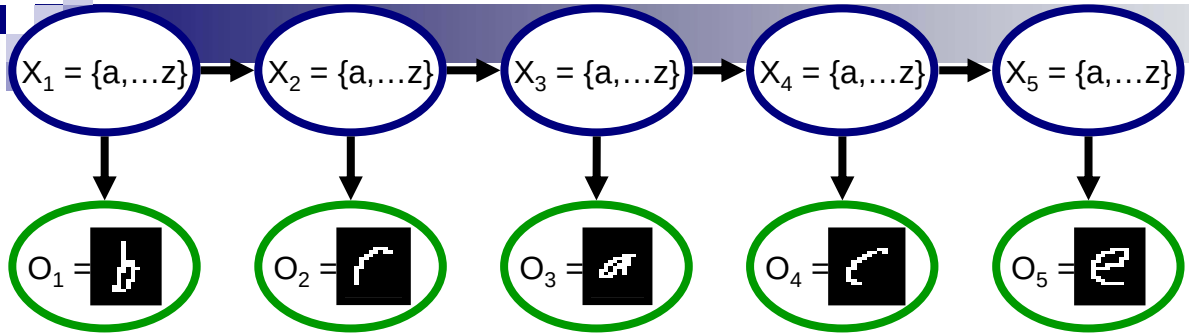
Compute:

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

Variable elimination order?

Example:

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



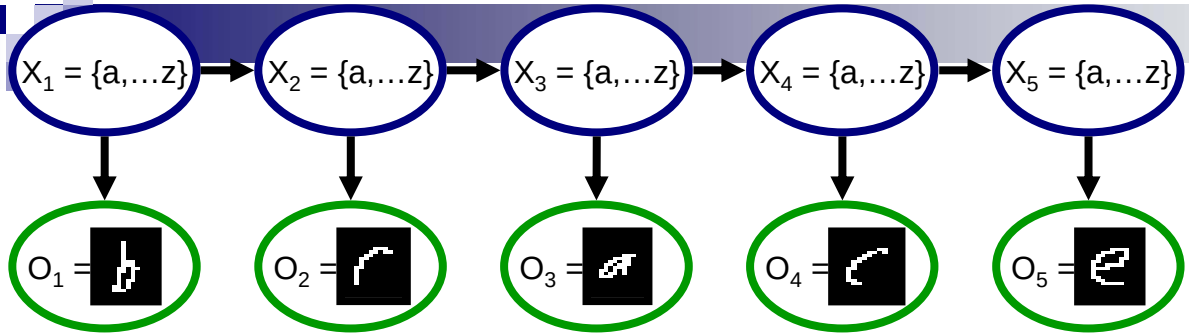
Compute:

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

Variable elimination for each i?

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

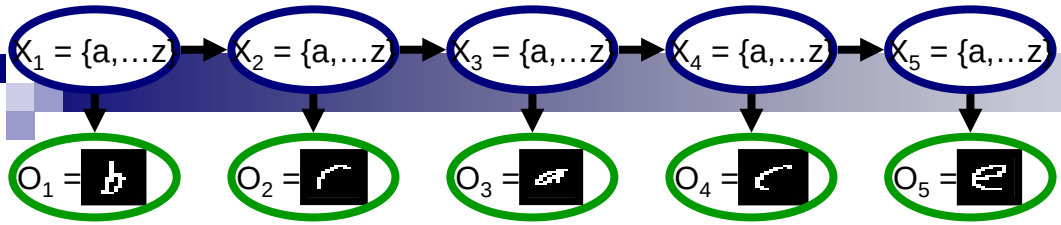
Reusing computation



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}

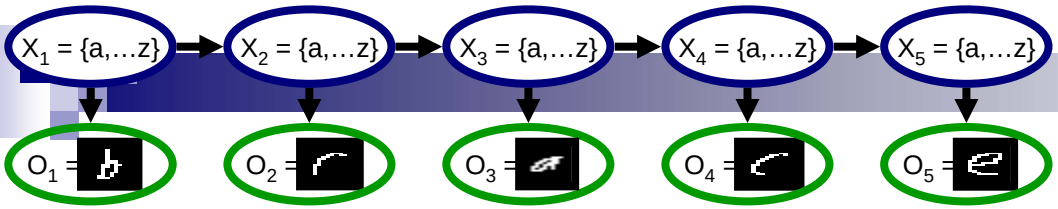
$$\alpha_i(X_i) = \sum_{x_{i-1}} P(o_i \mid X_i)P(X_i \mid X_{i-1} = x_{i-1})\alpha_{i-1}(x_{i-1})$$

- 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: $\boxed{P(X_i \mid o_{1..n}) = \alpha_i(X_i)\beta_i(X_i)}$

Most likely explanation

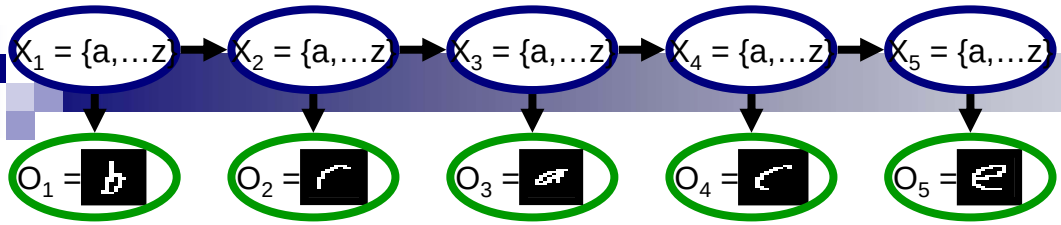


Compute:

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

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

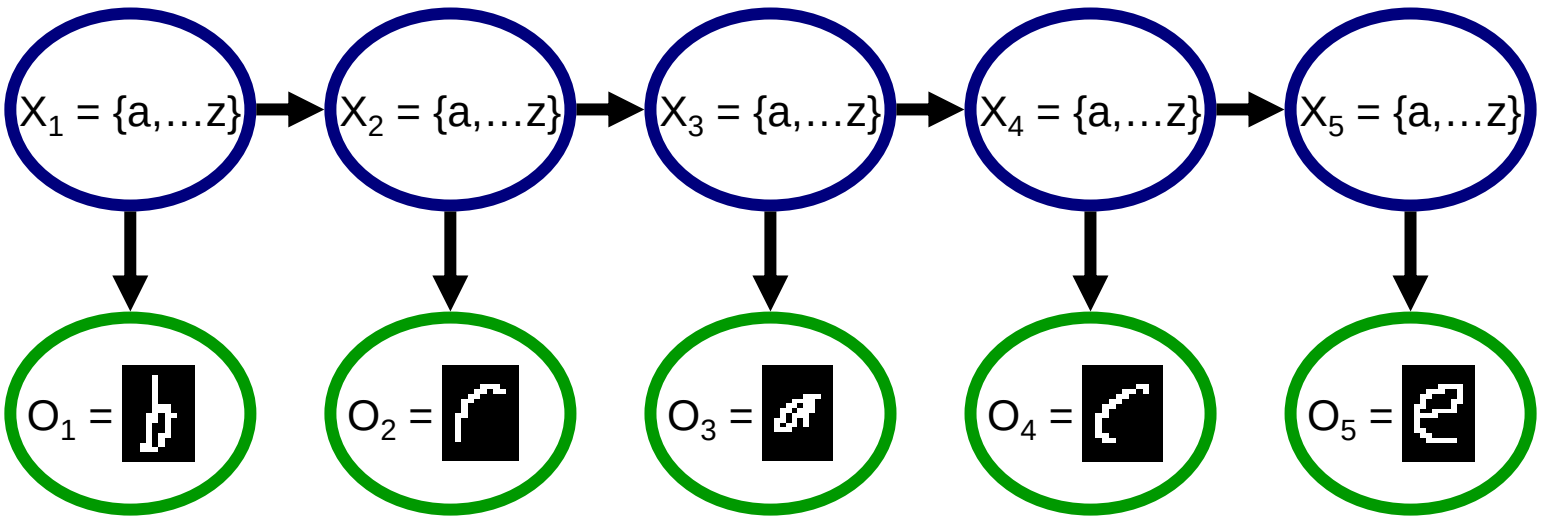
What you'll implement 1: multiplication

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

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