



EM for Bayes Nets

Machine Learning – 10701/15781

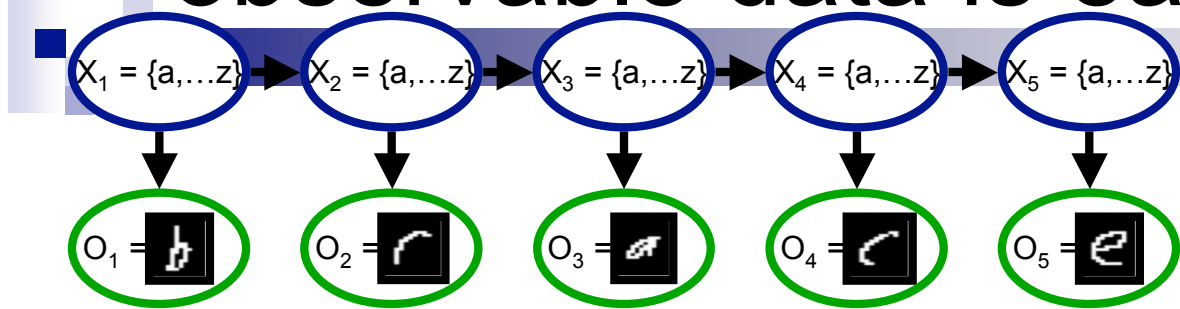
Carlos Guestrin

Carnegie Mellon University

April 16th, 2007

©2005-2007 Carlos Guestrin

Learning HMMs from fully observable data is easy



Learn 3 distributions:

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

$$P(O_i^{\text{pixel 17 is white}} | X_i^a) = \frac{\text{count}(\text{pixel 17 was white, } X_i = a)}{N_i}$$

$$P(X_i^a | X_{i-1}^b)$$

select training data where letter was a

What if O is observed,
but X is hidden

Log likelihood for HMMs when \mathbf{X} is hidden

$$\mathbf{o} = (o_1, \dots, o_n)$$

$$\mathbf{x} = (x_1, \dots, x_n)$$

for m sequences
 $\sum_{j=1}^m \log P(\mathbf{o}^{(j)} | \theta)$

■ Marginal likelihood – \mathbf{o} is observed, \mathbf{x} is missing

□ For simplicity of notation, training data consists of only one sequence:

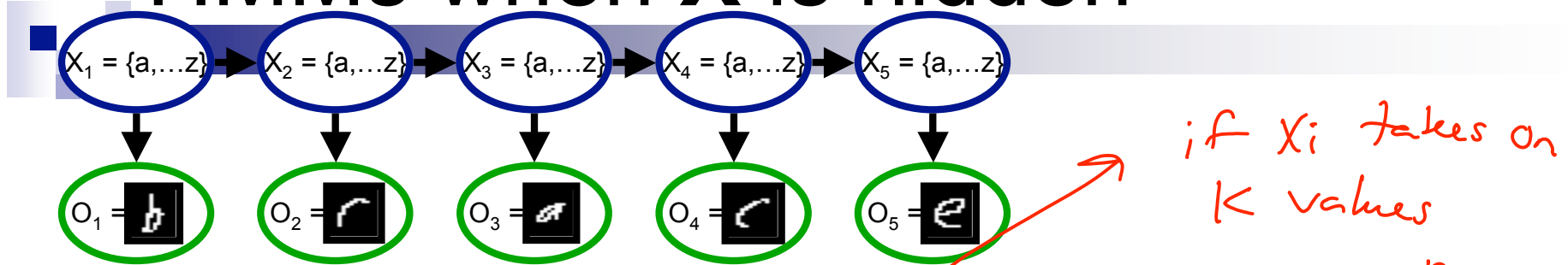
✓ *Observed*

$$\begin{aligned} \ell(\theta : \mathcal{D}) &= \log P(\mathbf{o} | \theta) \\ &= \log \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{o} | \theta) \end{aligned}$$

□ If there were m sequences:

$$\ell(\theta : \mathcal{D}) = \sum_{j=1}^m \log \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{o}^{(j)} | \theta)$$

Computing Log likelihood for HMMs when \mathbf{X} is hidden



if X_i takes on K values

Sum over K^n assignments

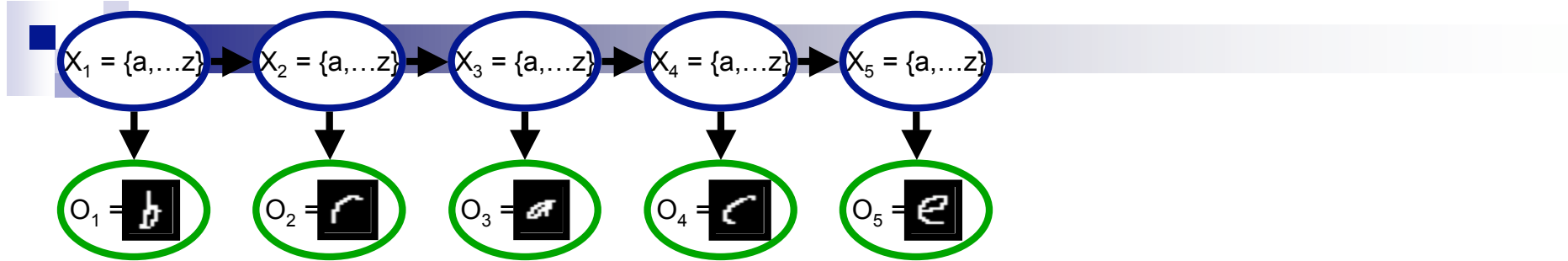
$$\begin{aligned} \ell(\theta : \mathcal{D}) &= \log P(\mathbf{o} | \theta) \\ &= \log \sum_{\mathbf{x}} P(\mathbf{x}, \mathbf{o} | \theta) \end{aligned}$$

$$= \log \sum_{x_1} \sum_{x_2} \dots \sum_{x_n} P(x_1) \cdot P(o_1 | x_1) \cdot \prod_{i=2}^n P(x_i | x_{i-1}) \cdot P(o_i | x_i)$$

$$= \log \sum_{x_1} \sum_{x_{n-1}} P(x_1) P(o_1 | x_1) \prod_{i=2}^{n-1} P(x_i | x_{i-1}) P(o_i | x_i) \underbrace{\sum_{x_n} P(x_n | x_{n-1}) \cdot P(o_n | x_n)}_{B_{n-1}(x_{n-1})}$$

use V.E. to
compute $\ell(\theta : \mathcal{D})$ in $O(n)$ time

The M-step



■ Maximization step:

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta} \sum_{\mathbf{x}} Q^{(t+1)}(\mathbf{x} | \mathbf{o}) \log P(\mathbf{x}, \mathbf{o} | \theta)$$

weighted log likelihood

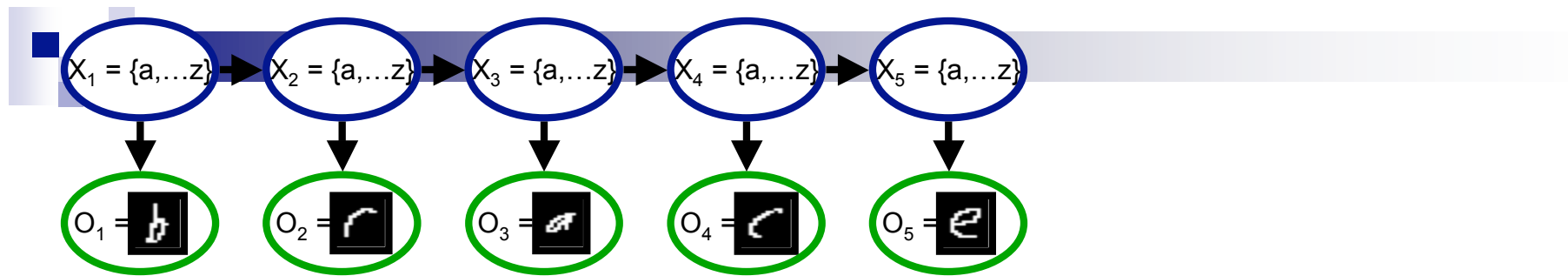
■ Use expected counts instead of counts:

- ☐ If learning requires Count(\mathbf{x}, \mathbf{o})
- ☐ Use $E_{Q^{(t+1)}}[\text{Count}(\mathbf{x}, \mathbf{o})]$

$$E_{Q^{(t+1)}}[\text{Count}(\mathbf{x} = \{a, b, c\}, \mathbf{o} = [\boxed{a}, \boxed{b}, \boxed{a}])] = \frac{\sum_{j=1}^m Q^{(t+1)}(\mathbf{x} = \{a, b, c\} | \mathbf{o} = [\boxed{a}, \boxed{b}, \boxed{a}])}{m}$$

E-step revisited

$$Q^{(t+1)}(\mathbf{x} | \mathbf{o}) = P(\mathbf{x} | \mathbf{o}, \theta^{(t)})$$



- E-step computes probability of hidden vars \mathbf{x} given \mathbf{o}

- Must compute:

- $Q(x_t=a|\mathbf{o})$ – marginal probability of each position

- Just forwards-backwards!

- $Q(x_{t+1}=a, x_t=b|\mathbf{o})$ – joint distribution between pairs of positions

see reading

[simple eqn.]

[maybe homework]

Exploiting unlabeled data in clustering

- A few data points are labeled

- $\langle x, o \rangle$

- Most points are unlabeled

- $\langle ?, o \rangle$

- In the E-step of EM:

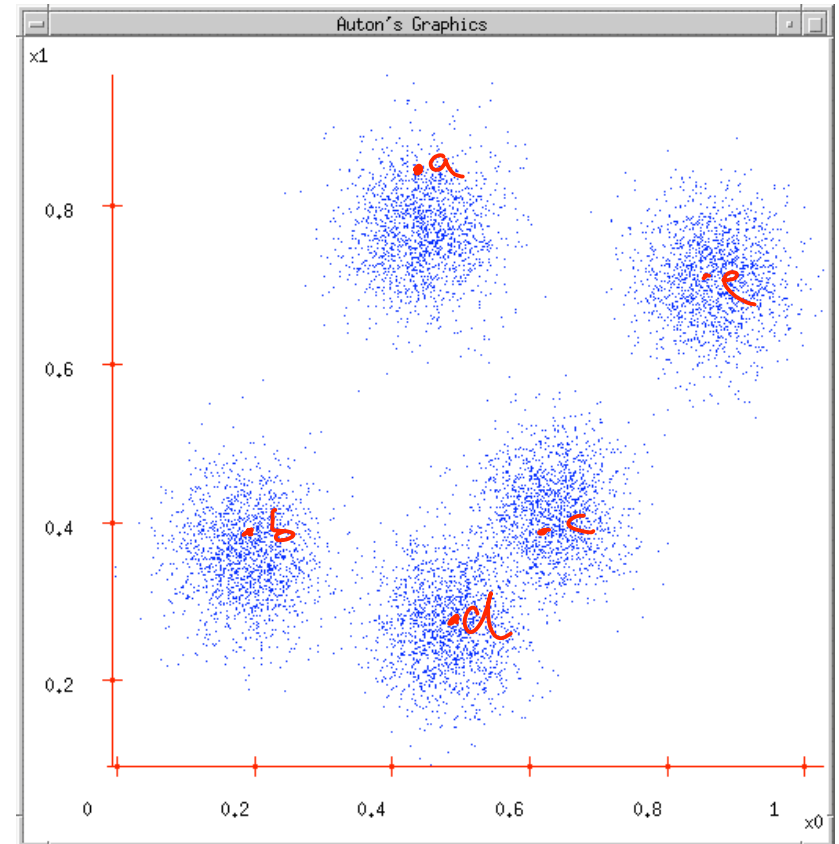
- If i'th point is unlabeled:

- compute $Q(X|o_i)$ as usual

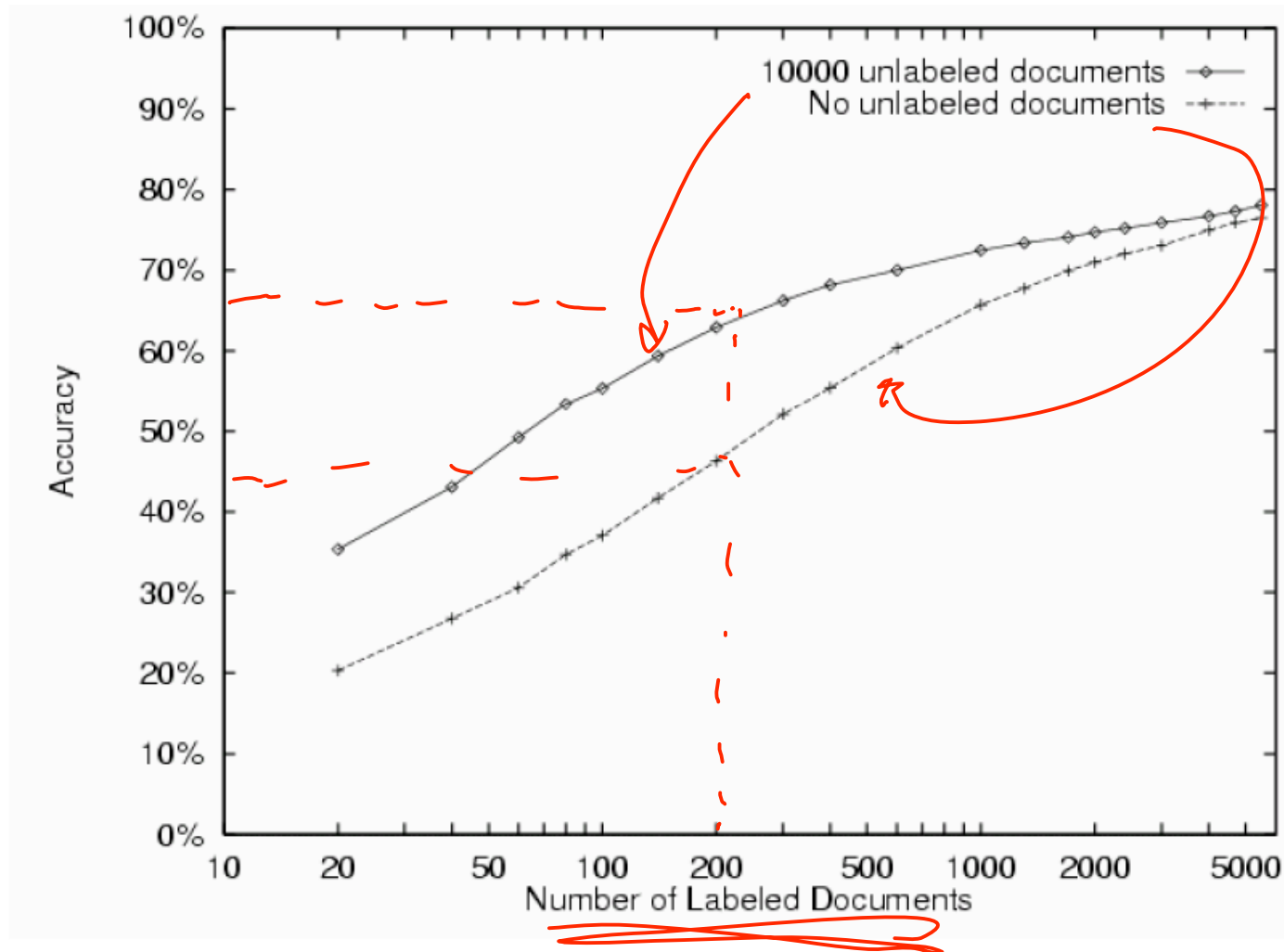
- If i'th point is labeled:

- set $Q(X=x|o_i)=1$ and $Q(X \neq x|o_i)=0$

- M-step as ~~usual~~ ^{correct label}



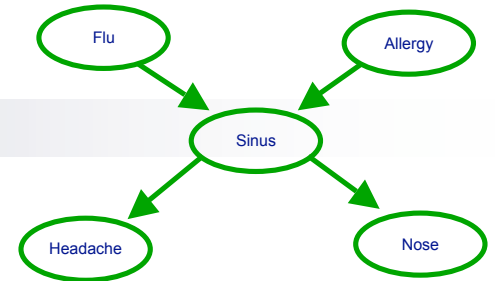
20 Newsgroups data – advantage of adding unlabeled data



Data likelihood for BNs

- Given structure, log likelihood of fully observed data:

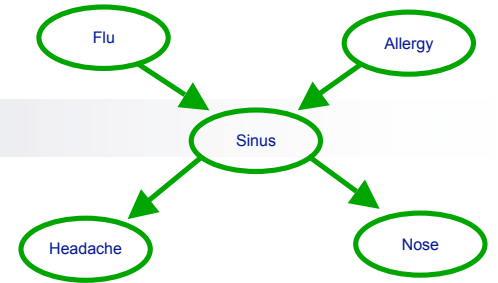
$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G})$$



Marginal likelihood

- What if S is hidden?

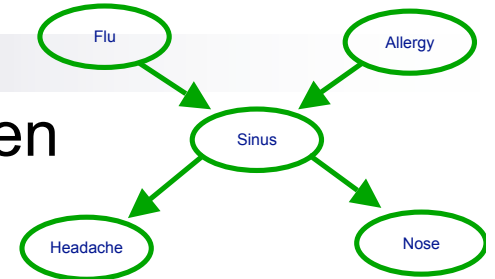
$$\log P(\mathcal{D} \mid \theta_{\mathcal{G}}, \mathcal{G})$$



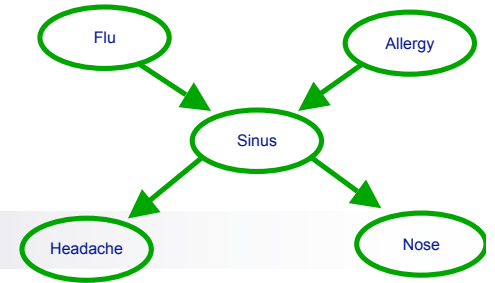
Log likelihood for BNs with hidden data

- Marginal likelihood – **O** is observed, **H** is hidden

$$\begin{aligned}\ell(\theta : \mathcal{D}) &= \sum_{j=1}^m \log P(\mathbf{o}^{(j)} \mid \theta) \\ &= \sum_{j=1}^m \log \sum_{\mathbf{h}} P(\mathbf{h}, \mathbf{o}^{(j)} \mid \theta)\end{aligned}$$



E-step for BNs

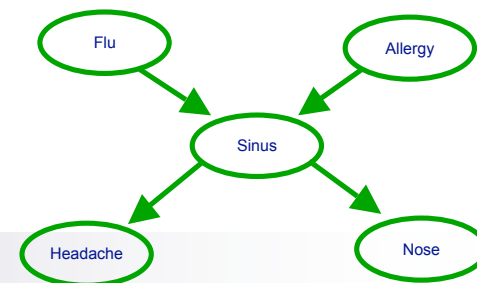


- E-step computes probability of hidden vars **h** given **o**

$$Q^{(t+1)}(\mathbf{h} \mid \mathbf{o}) = P(\mathbf{h} \mid \mathbf{o}, \theta^{(t)})$$

- Corresponds to inference in BN

The M-step for BNs



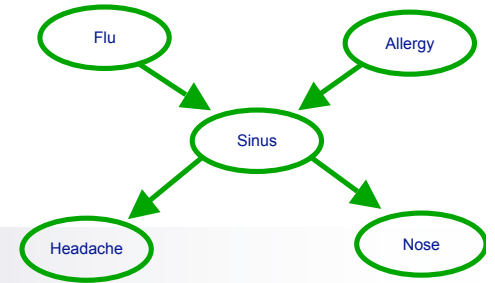
- Maximization step:

$$\theta^{(t+1)} \leftarrow \arg \max_{\theta} \sum_{\mathbf{x}} Q^{(t+1)}(\mathbf{h} \mid \mathbf{o}) \log P(\mathbf{h}, \mathbf{o} \mid \theta)$$

- Use expected counts instead of counts:

- ☐ If learning requires $\text{Count}(\mathbf{h}, \mathbf{o})$
- ☐ Use $E_{Q^{(t+1)}}[\text{Count}(\mathbf{h}, \mathbf{o})]$

M-step for each CPT



- M-step decomposes per CPT

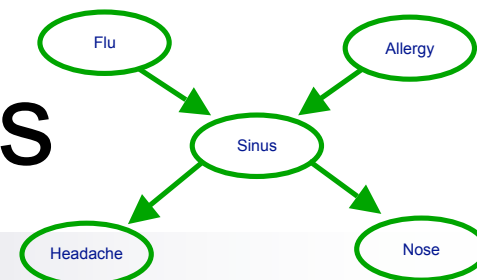
- Standard MLE:

$$P(X_i = x_i \mid \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{Count}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{Count}(\text{Pa}_{X_i} = \mathbf{z})}$$

- M-step uses expected counts:

$$P(X_i = x_i \mid \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{ExCount}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{ExCount}(\text{Pa}_{X_i} = \mathbf{z})}$$

Computing expected counts



$$P(X_i = x_i \mid \text{Pa}_{X_i} = \mathbf{z}) = \frac{\text{ExCount}(X_i = x_i, \text{Pa}_{X_i} = \mathbf{z})}{\text{ExCount}(\text{Pa}_{X_i} = \mathbf{z})}$$

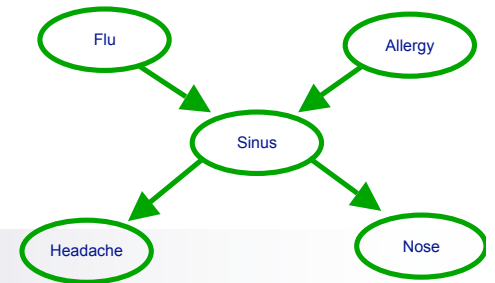
■ M-step requires expected counts:

- For a set of vars **A**, must compute $\text{ExCount}(\mathbf{A}=\mathbf{a})$
- Some of **A** in example j will be observed
 - denote by $\mathbf{A}_O = \mathbf{a}_O^{(j)}$
- Some of **A** will be hidden
 - denote by \mathbf{A}_H

■ Use inference (E-step computes expected counts):

- $\text{ExCount}^{(t+1)}(\mathbf{A}_O = \mathbf{a}_O^{(j)}, \mathbf{A}_H = \mathbf{a}_H) \leftarrow P(\mathbf{A}_H = \mathbf{a}_H, \mathbf{A}_O = \mathbf{a}_O^{(j)} \mid \theta^{(t)})$

Data need not be hidden in the same way



- When data is fully observed
 - A data point is
- When data is partially observed
 - A data point is
- But unobserved variables can be different for different data points
 - e.g.,
- Same framework, just change definition of expected counts
 - $\text{ExCount}^{(t+1)}(\mathbf{A}_O = \mathbf{a}_O^{(j)}, \mathbf{A}_H = \mathbf{a}_H) \leftarrow P(\mathbf{A}_H = \mathbf{a}_H, \mathbf{A}_O = \mathbf{a}_O^{(j)} | \theta^{(t)})$

What you need to know

- EM for Bayes Nets
- E-step: inference computes expected counts
 - Only need expected counts over X_i and \mathbf{Pa}_{x_i}
- M-step: expected counts used to estimate parameters
- Hidden variables can change per datapoint
- Use labeled and unlabeled data ! some data points are complete, some include hidden variables



Co-Training for Semi-supervised learning

Machine Learning – 10701/15781

Carlos Guestrin

Carnegie Mellon University

April 16th, 2007

©2005-2007 Carlos Guestrin

Redundant information

Professor Faloutsos

my advisor



U.S. mail address:
Department of Computer Science
University of Maryland
College Park, MD 20742
(97-99: [on leave at CMU](#))
Office: 3227 A.V. Williams Bldg.
Phone: (301) 405-2695
Fax: (301) 405-6707
Email: christos@cs.umd.edu

Christos Faloutsos

Current Position: Assoc. Professor of [Computer Science](#). (97-98: [on leave at CMU](#))

Join Appointment: [Institute for Systems Research](#) (ISR).

Academic Degrees: Ph.D. and M.Sc. ([University of Toronto](#)), B.Sc. ([Nat. Tech. U. Ath](#))

Research Interests:

- Query by content in multimedia databases;
- Fractals for clustering and spatial access methods;
- Data mining;

Redundant information – webpage text

**U.S. mail address:**

Department of Computer Science
University of Maryland
College Park, MD 20742
(97-99: on leave at CMU)

Office: 3227 A.V. Williams Bldg.

Phone: (301) 405-2695

Fax: (301) 405-6707

Email: christos@cs.umd.edu

Christos Faloutsos

Current Position: Assoc. Professor of [Computer Science](#). (97-98: on leave at CMU)

Join Appointment: [Institute for Systems Research](#) (ISR).

Academic Degrees: Ph.D. and M.Sc. ([University of Toronto](#)), B.Sc. ([Nat. Tech. U. Ath](#))

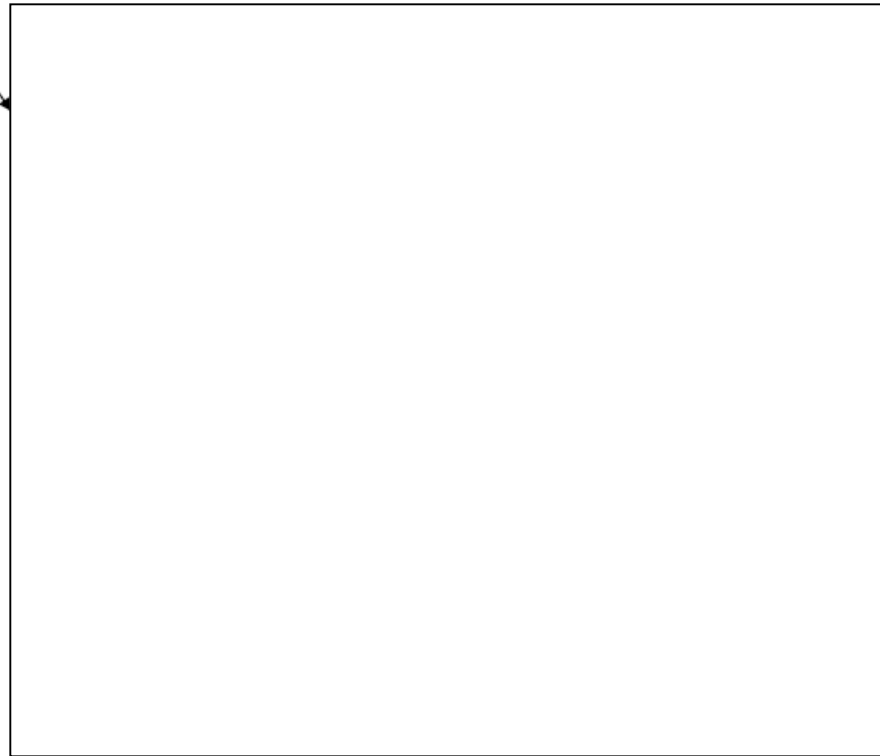
Research Interests:

- Query by content in multimedia databases;
- Fractals for clustering and spatial access methods;
- Data mining;

Redundant information – anchor text for hyperlinks

Professor Faloutsos

my advisor




Exploiting redundant information in semi-supervised learning

- Want to predict Y from features \mathbf{X}
 - $f(\mathbf{X}) \mapsto Y$
 - have some labeled data \mathbf{L}
 - lots of unlabeled data \mathbf{U}
- Co-training assumption: \mathbf{X} is very expressive
 - $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$
 - can learn
 - $g_1(\mathbf{X}_1) \mapsto Y$
 - $g_2(\mathbf{X}_2) \mapsto Y$

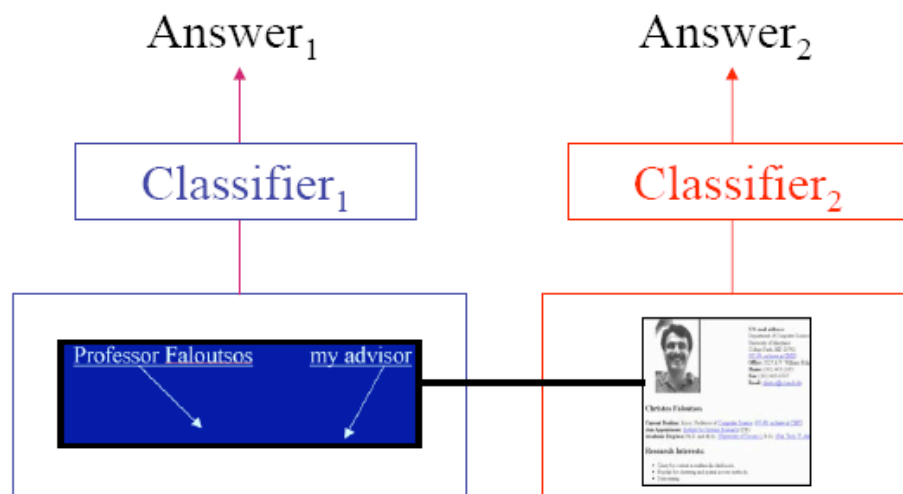
Professor Faloutsos

my advisor

	<p>U.S. mail address: Department of Computer Science University of Maryland College Park, MD 20742 (97-99: on leave at CMU) Office: 3227 A.V. Williams Bldg. Phone: (301) 405-2695 Fax: (301) 405-6707 Email: christos@cs.umd.edu</p>
<p>Christos Faloutsos</p> <p>Current Position: Assoc. Professor of Computer Science, (97-99: on leave at CMU) Join Appointment: Institute for Systems Research (ISR). Academic Degrees: Ph.D. and M.Sc. (University of Toronto), B.Sc. (Nat. Tech. U. Athens)</p> <p>Research Interests:</p> <ul style="list-style-type: none">• Query by content in multimedia databases;• Fractals for clustering and spatial access methods;• Data mining.	

Co-Training

- Key idea: Classifier₁ and Classifier₂ must:
 - Correctly classify labeled data
 - **Agree** on unlabeled data



Co-Training Algorithm

[Blum & Mitchell '99]

Given: labeled data L ,

unlabeled data U

Loop:

Train g_1 (hyperlink classifier) using L

Train g_2 (page classifier) using L

Allow g_1 to label p positive, n negative examps from U

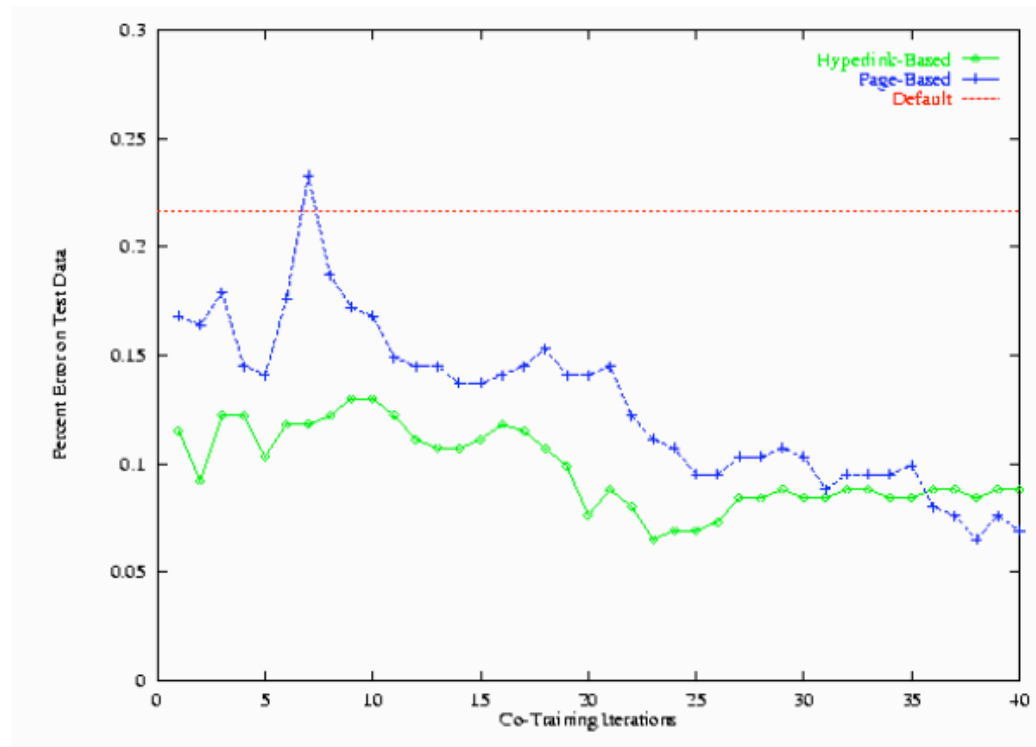
Allow g_2 to label p positive, n negative examps from U

Add these self-labeled examples to L

Co-Training experimental results

- begin with 12 labeled web pages (academic course)
- provide 1,000 additional unlabeled web pages
- average error: learning from labeled data 11.1%;
- average error: cotraining 5.0%

Typical run:



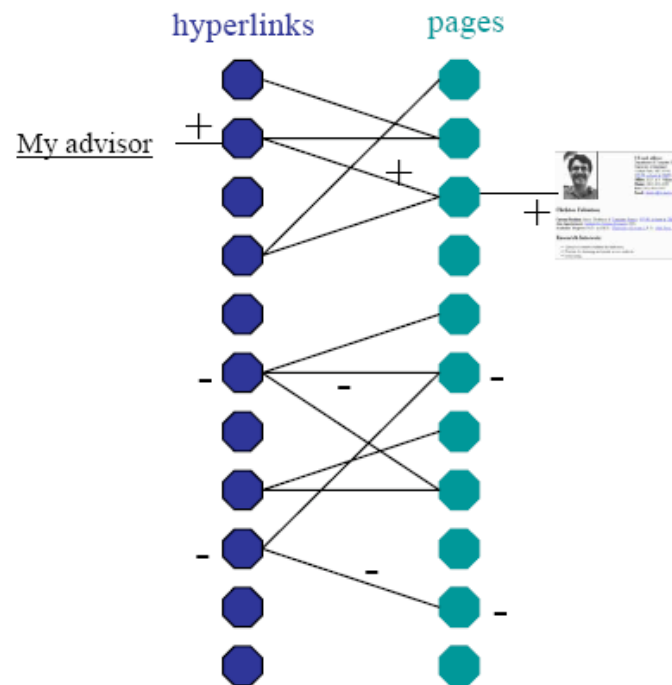
Co-Training theory

- Want to predict Y from features \mathbf{X}
 - $f(\mathbf{X}) \mapsto Y$
- Co-training assumption: \mathbf{X} is very expressive
 - $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$
 - want to learn $g_1(\mathbf{X}_1) \mapsto Y$ and $g_2(\mathbf{X}_2) \mapsto Y$
- *Assumption:* $\exists g_1, g_2, \forall \mathbf{x} \ g_1(\mathbf{x}_1) = f(\mathbf{x}), g_2(\mathbf{x}_2) = f(\mathbf{x})$
- Questions:
 - Does unlabeled data always help?
 - How many labeled examples do I need?
 - How many unlabeled examples do I need?

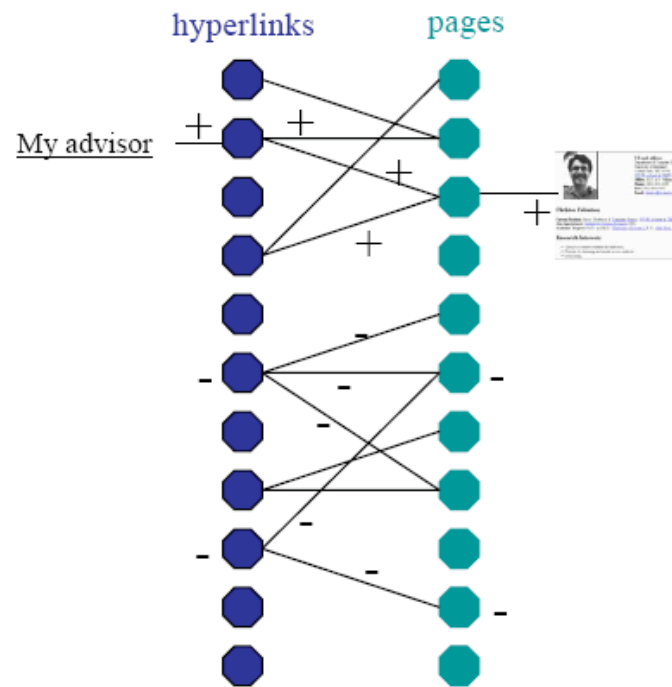
Understanding Co-Training: A simple setting

- Suppose \mathbf{X}_1 and \mathbf{X}_2 are discrete
 - $|\mathbf{X}_1| = |\mathbf{X}_2| = N$
- No label noise
- Without unlabeled data, how hard is it to learn g_1 (or g_2)?

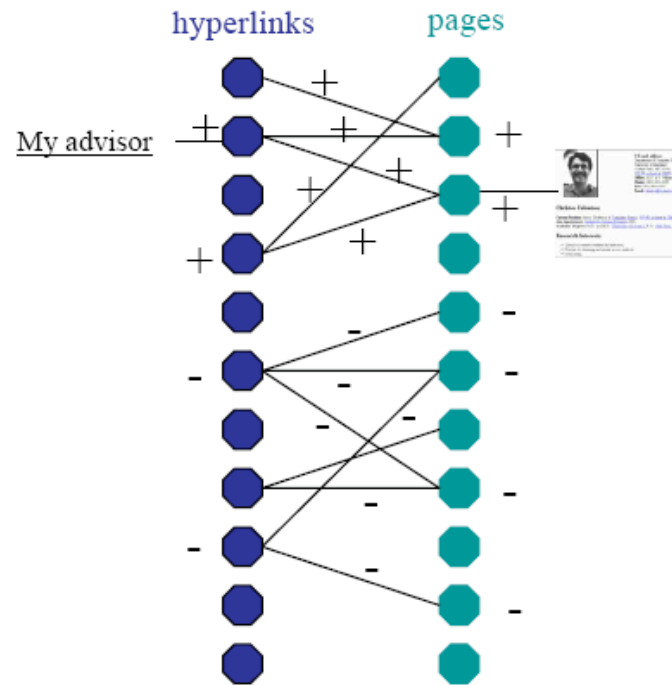
Co-Training in simple setting – Iteration 0



Co-Training in simple setting – Iteration 1

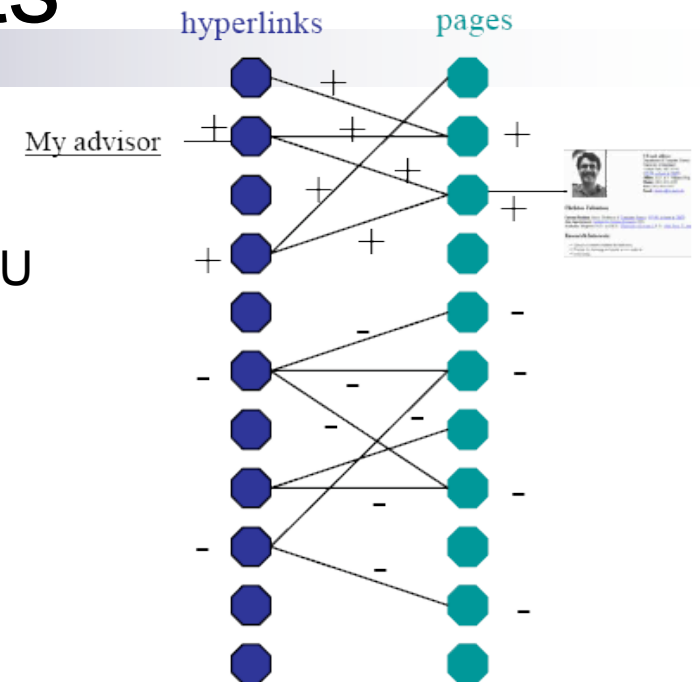


Co-Training in simple setting – after convergence



Co-Training in simple setting – Connected components

- Suppose infinite **unlabeled** data
 - Co-training must have at least one labeled example in each connected component of L+U graph
- What's probability of making an error?
- For k Connected components, how much labeled data?

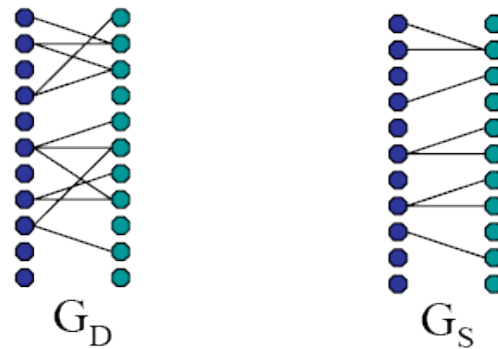


$$E[error] = \sum_j P(x \in g_j)(1 - P(x \in g_j))^m$$

Where g_j is the j th connected component of graph of L+U, m is number of labeled examples

How much unlabeled data?

Want to assure that connected components in the underlying distribution, G_D , are connected components in the observed sample, G_S



$O(\log(N)/\alpha)$ examples assure that with high probability, G_S has same connected components as G_D [Karger, 94]

N is size of G_D , α is min cut over all connected components of G_D

Co-Training theory

- Want to predict Y from features \mathbf{X}
 - $f(\mathbf{X}) \mapsto Y$
- Co-training assumption: \mathbf{X} is very expressive
 - $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2)$
 - want to learn $g_1(\mathbf{X}_1) \mapsto Y$ and $g_2(\mathbf{X}_2) \mapsto Y$
- *Assumption:* $\exists g_1, g_2, \forall \mathbf{x} \ g_1(\mathbf{x}_1) = f(\mathbf{x}), g_2(\mathbf{x}_2) = f(\mathbf{x})$
- One co-training result [Blum & Mitchell '99]
 - If
 - $(\mathbf{X}_1 \perp \mathbf{X}_2 \mid Y)$
 - g_1 & g_2 are PAC learnable from noisy data (and thus f)
 - Then
 - f is PAC learnable from weak initial classifier plus unlabeled data

What you need to know about co-training

- Unlabeled data can help supervised learning (a lot) when there are (mostly) independent redundant features
- One theoretical result:
 - If $(\mathbf{X}_1 \perp \mathbf{X}_2 \mid Y)$ and g_1 & g_2 are PAC learnable from noisy data (and thus f)
 - Then f is PAC learnable from weak initial classifier plus unlabeled data
 - Disagreement between g_1 and g_2 provides bound on error of final classifier
- Applied in many real-world settings:
 - Semantic lexicon generation [Riloff, Jones 99] [Collins, Singer 99], [Jones 05]
 - Web page classification [Blum, Mitchell 99]
 - Word sense disambiguation [Yarowsky 95]
 - Speech recognition [de Sa, Ballard 98]
 - Visual classification of cars [Levin, Viola, Freund 03]

Acknowledgement



- I would like to thank Tom Mitchell for some of the material used in this presentation of co-training