Sequential Learning

WHAT IS SEQUENTIAL LEARNING?

Topics from class

- Classification learning: learn $\mathbf{x} \rightarrow y$
  - Linear (naïve Bayes, logistic regression, …)
  - Nonlinear (neural nets, trees, …)
- Not quite classification learning:
  - Regression (y is a number)
  - Clustering, EM, graphical models, …
    - there is no y, so build a distributional model the instances
  - Collaborative filtering/matrix factoring
    - many linked regression problems
  - **Learning for sequences:** learn $(\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_k) \rightarrow (y_1,\ldots,y_k)$
    - special case of "**structured output prediction**"

A sequence learning task: named entity recognition (NER)

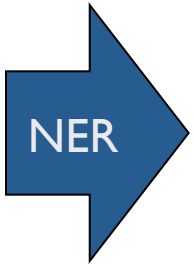
October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying…



person company jobTitle

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying…

Name entity recognition (NER) is one part of information extraction (IE)

person company jobTitle

NER

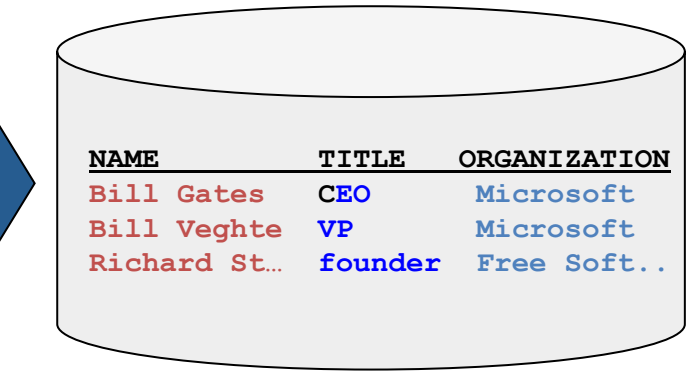
October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

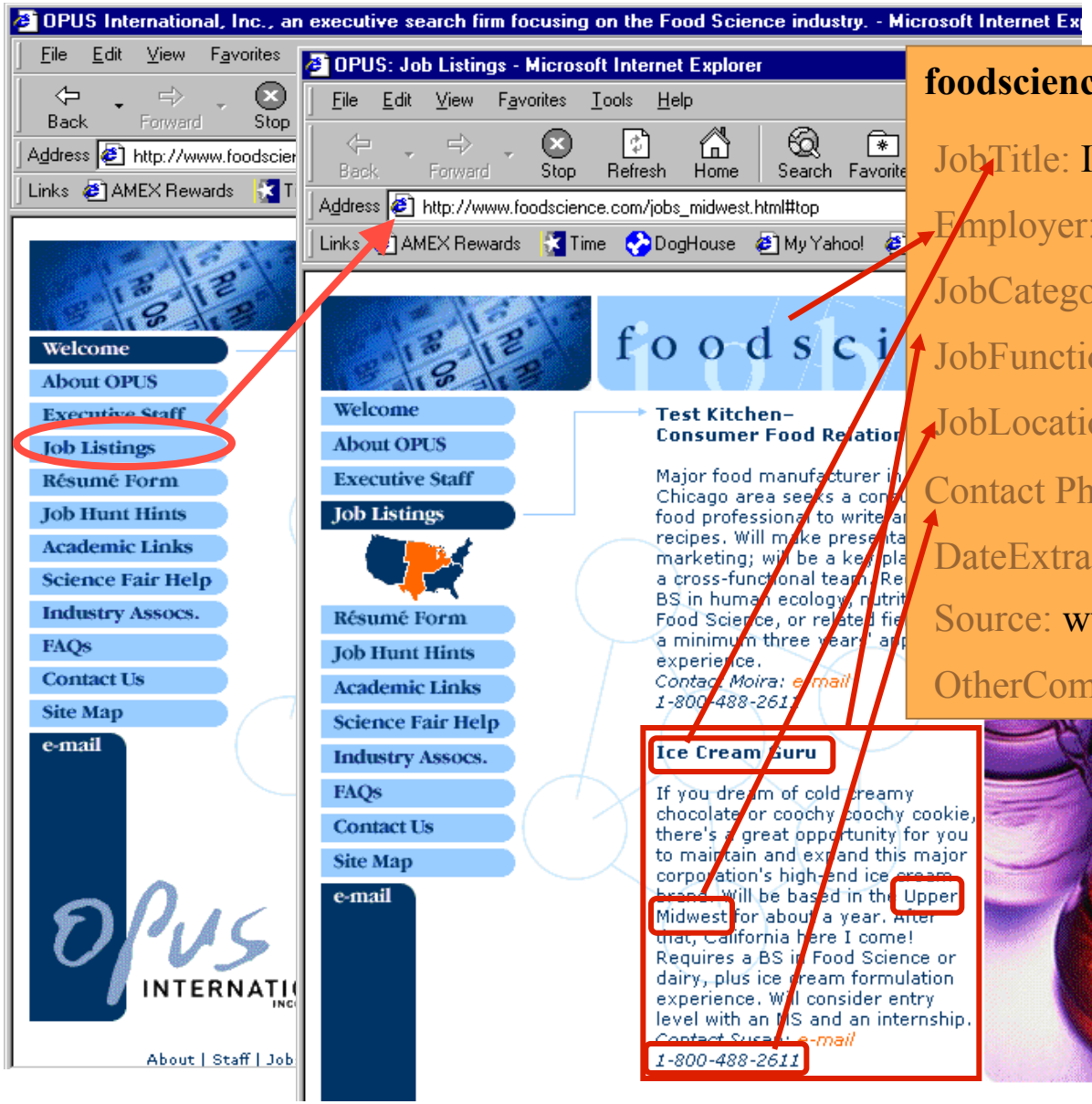
Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying…



IE Example:Job Openings from the Web



foodscience.com-Job2

JobTitle: Ice Cream Guru

Employer: foodscience.com

JobCategory: Travel/Hospitality

JobFunction: Food Services

JobLocation: Upper Midwest

Contact Phone: 800-488-2611

DateExtracted: January 8, 2001

Source: www.foodscience.com/jobs_midwest.htm

OtherCompanyJobs: foodscience.com-Job1

IE Example: A Job Search Site

How can we do NER?

person company jobTitle

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

Richard Stallman, founder of the Free Software Foundation, countered saying…

October 14, 2002, 4:00 a.m. PT

For years, Microsoft Corporation CEO Bill Gates railed against the economic philosophy of open-source software with Orwellian fervor, denouncing its communal licensing as a "cancer" that stifled technological innovation.

Today, Microsoft claims to "love" the open-source concept, by which software code is made public to encourage improvement and development by outside programmers. Gates himself says Microsoft will gladly disclose its crown jewels--the coveted code behind the Windows operating system--to select customers.

"We can be open source. We love the concept of shared source," said Bill Veghte, a Microsoft VP. "That's a super-important shift for us in terms of code access."

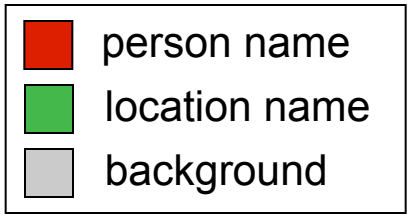
Richard Stallman, founder of the Free Software Foundation, countered saying…

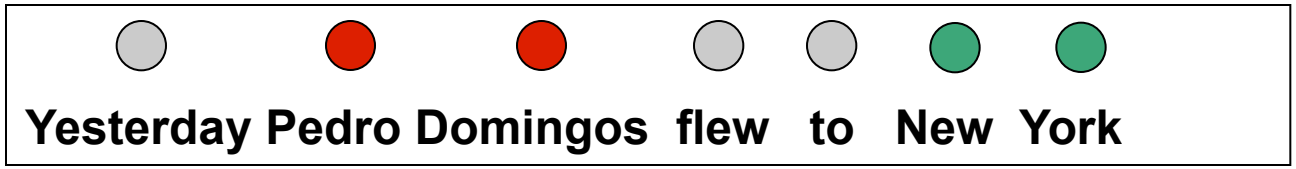
Most common approach: NER by classifying tokens

Given a sentence:

Yesterday Pedro Domingos flew to New York.

1) Break the sentence into *tokens*, and ***classify*** each token with a label indicating *what sort of entity* it is part of:





2) Identify names based on the entity labels

Person name: **Pedro Domingos**
Location name: **New York**

3) To learn an NER system, use YFCL and whatever features you want….

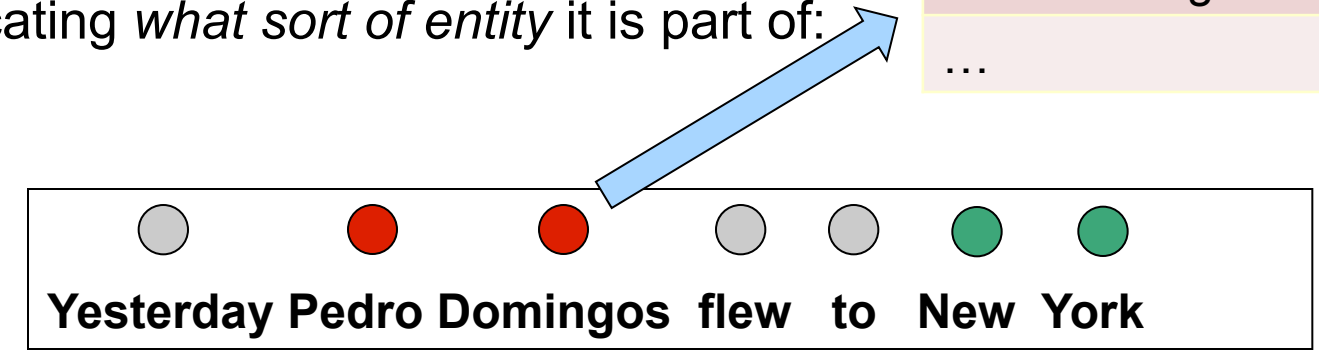
Most common approach: NER by classifying tokens

Given a sentence:

Yesterday Pedro Domingos flew to New

Feature	Value
isCapitalized	yes
numLetters	8
suffix2	-os
word-1-to-right	flew
word-2-to-right	to
…	

1) Break the sentence into *tokens*, and ***classify*** each token with a label indicating *what sort of entity* it is part of:



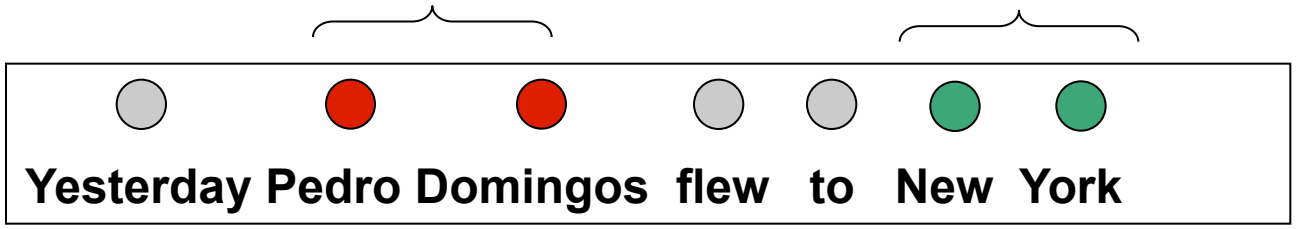
2) Identify names based on the entity labels

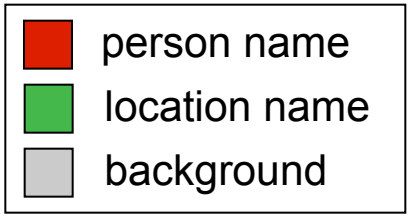
Person name: Pedro Domingos
Location name: New York

3) To learn an NER system, use YFCL and whatever features you want….

NER by classifying tokens

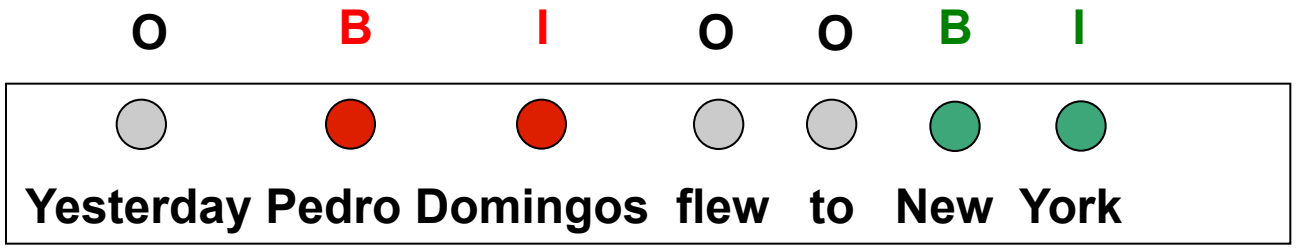
A Problem/Opportunity: YFCL assumes examples are iid.
But similar labels tend to *cluster together* in text

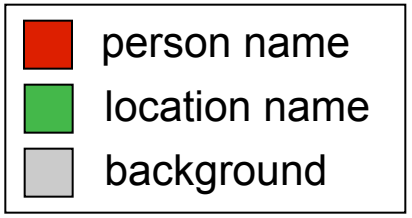




How can you model these dependencies?

NER by classifying tokens





Another common labeling scheme is **BIO** (begin, inside, outside; e.g. beginPerson, insidePerson, beginLocation, insideLocation, outside)

- **Begin** tokens are different from other name tokens
- "**Tell William Travis is handling it**"

BIO also leads to *strong dependencies between nearby labels* (eg **inside** follows **begin**).

How can you model these dependencies?

A **hidden Markov model (HMM):** the "naïve Bayes" of sequences

Other nice problems for HMMS

Parsing addresses

House number	Building	Road	City	State	Zip
4089	Whispering Pines	Nobel Drive	San Diego	CA	92122

Parsing citations

Author Year

P.P.Wangikar, T.P. Graycar, D.A. Estell, D.S. Clark, J.S. Dordick (1993) Protein and Solvent Engineering of Subtilising BPN' in Nearly Anhydrous Organic Media J.Amer. Chem. Soc. 115, 12231-12237.

Title Journal Volume

Other nice problems for HMMS

Sentence segmentation: Finding words (to index) in Asian languages

第二阶段的奥运会体育比赛门票与残奥会开闭幕式门票的预订工作已经结束,现在进入门票分配阶段。在此期间,我们不再接受新的门票预订申请。

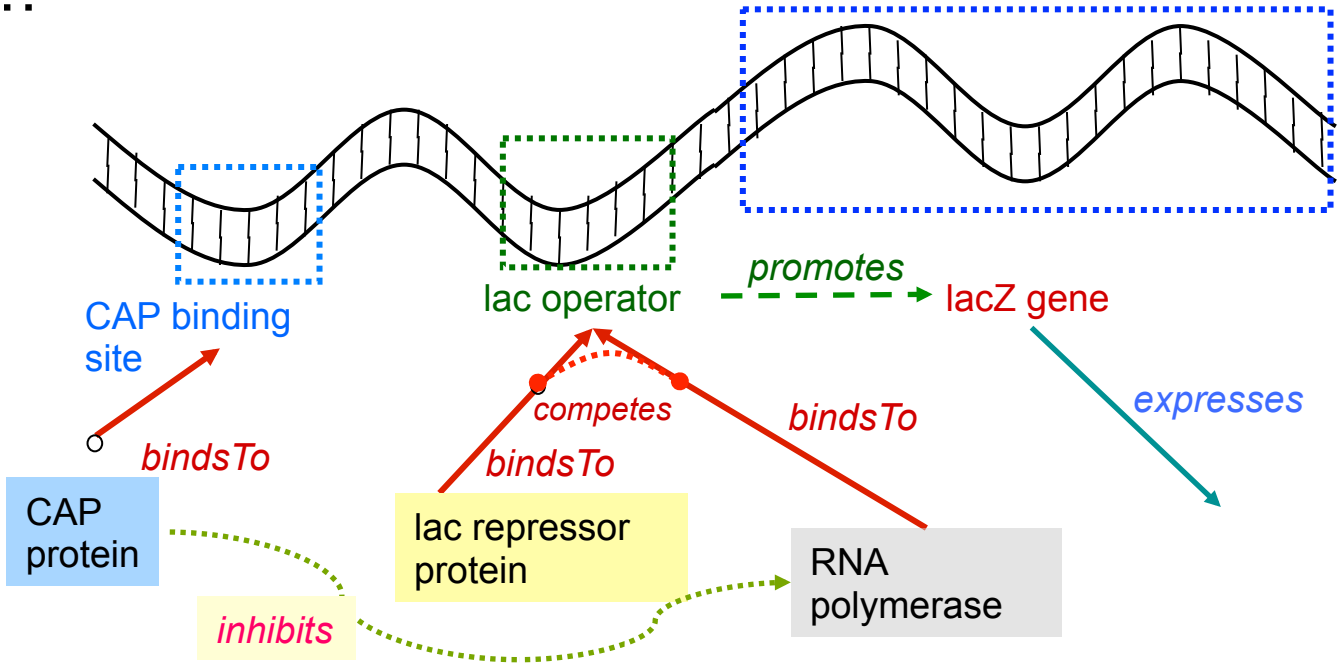
Morphology: Finding components of a single word

uygarlaştıramadıklarımızdanmışsınızcasına, or “(behaving) as if you are among those whom we could not civilize”

- Document analysis: finding tables in plain-text documents
- Video segmentation: splitting videos into naturally meaningful sections
- Converting text to speech (TTS)
- Converting speech to text (ASR)
- …

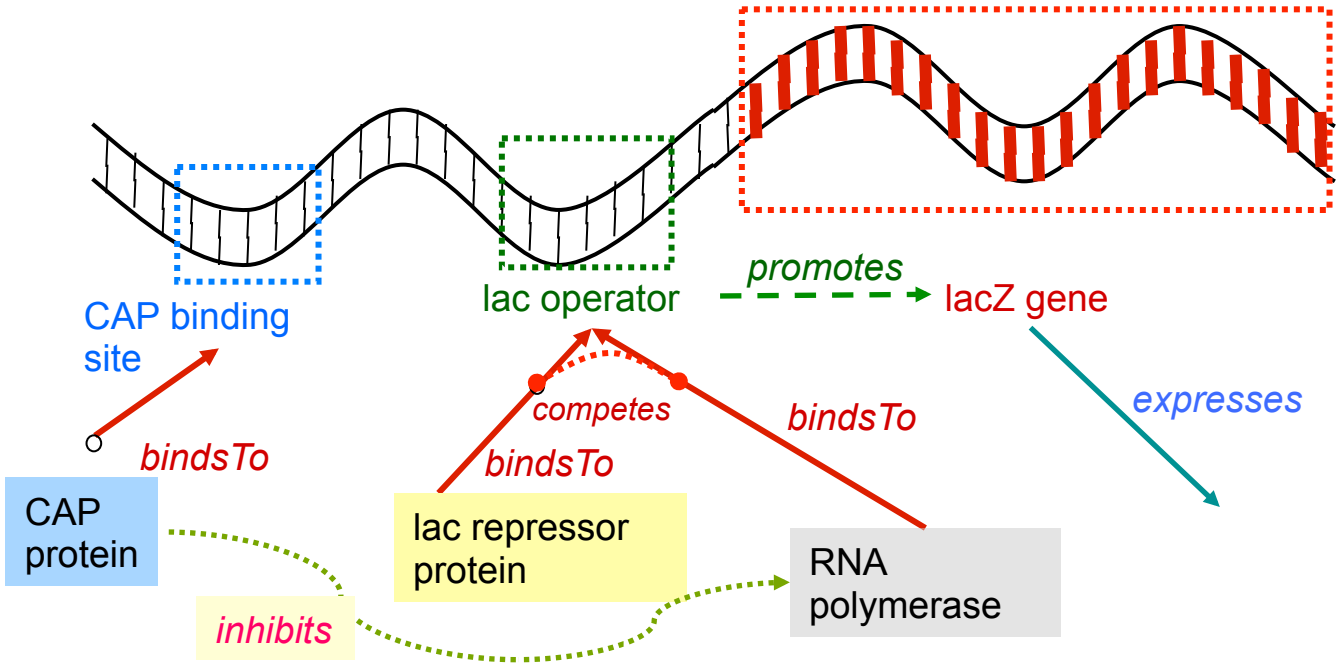
Other nice problems for HMMS

- Modeling **biological sequences**
  - e.g., segmenting DNA into genes (transcribed into proteins or not), promotors, TF binding sites, …
  - identifying variants of a single gene
  - …



Other nice problems for HMMS

- Eg **gene finding**: which parts of DNA are genes, versus binding sites for gene regulators, junk DNA, … ?

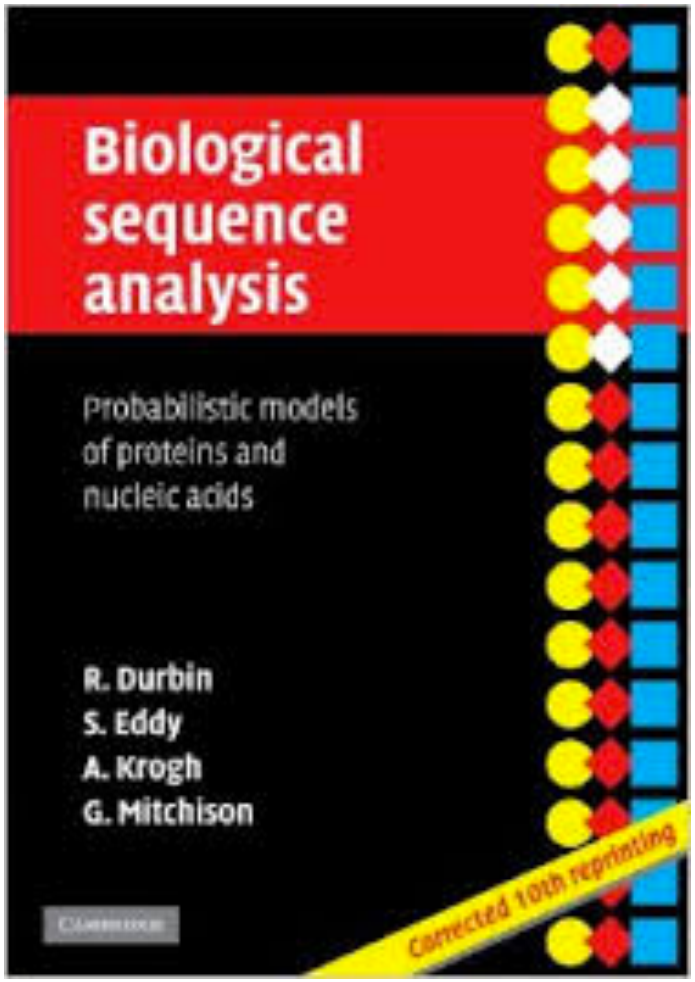


[BOOK] **Biological sequence analysis**: probabilistic models of proteins and nucleic acids

R Durbin, SR Eddy, A Krogh, G Mitchison - 1998 - books.google.com

Probabilistic models are becoming increasingly important in analysing the huge amount of data being produced by large-scale DNA-sequencing efforts such as the Human Genome Project. For example, hidden Markov models are used for analysing **biological** sequences, **...**

Cited by 5878 Related articles All 15 versions Cite Saved More



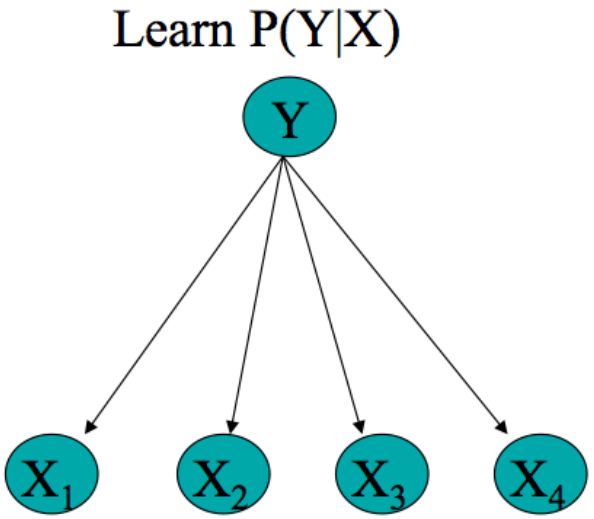
Aside: relax, we will not test you on biology for this class ☺

Sequence alignment for proteins (done by "pair HMMs")

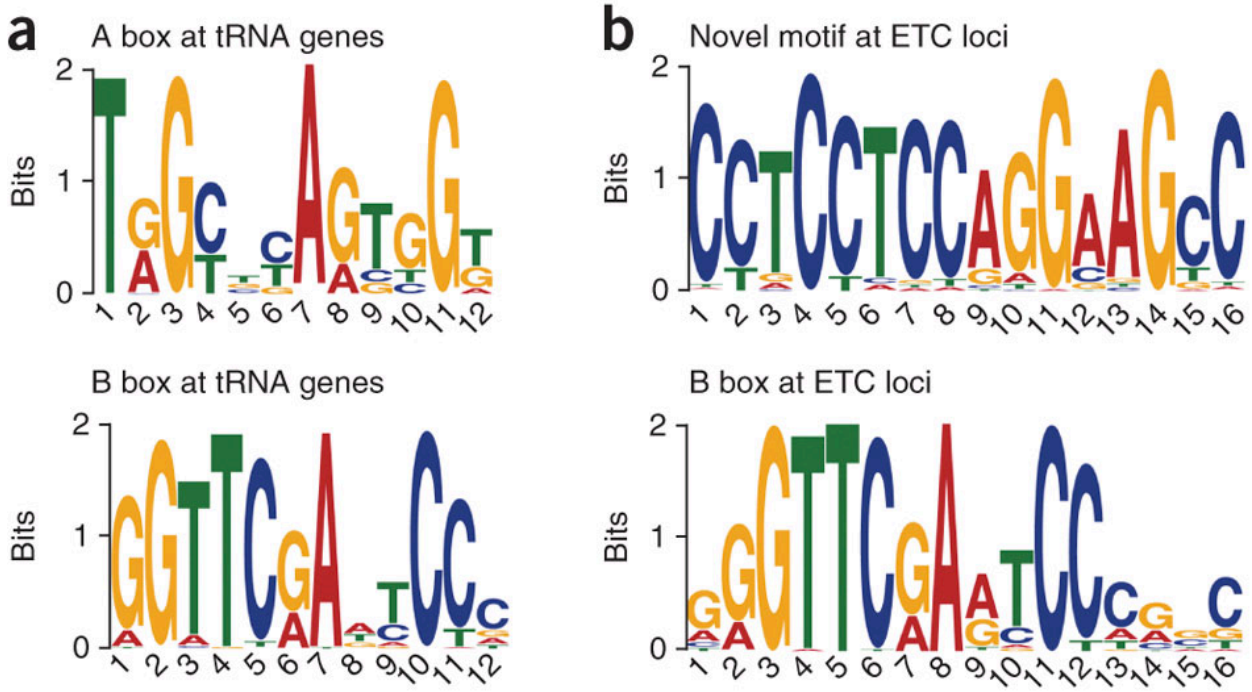
HMM warmup: a model of aligned sequences

```
GNA.GSAIG
SHA.GTCLG
SNA.GTAIG
SHA.GTCLG
GHA.GTCLG
GNA.GSALG
NNA.SITIG
NHA.GVALG
```

	S1	S2	S3	…
A	0.01	0.03	0.89	….
G	0.3	0.01	0.01	…
H	0.01	0.5	0.01	…
N	0.2	0.4	(	
S	0.3	0.01	(	
…				

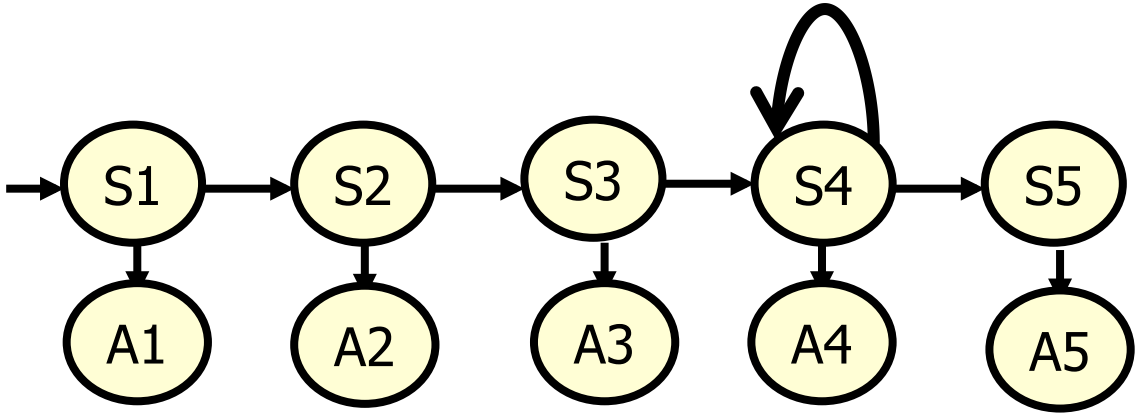


E.g.: Motifs


a
A box at tRNA genes
Bits
B box at tRNA genes
b
Novel motif at ETC loci
B box at ETC loci

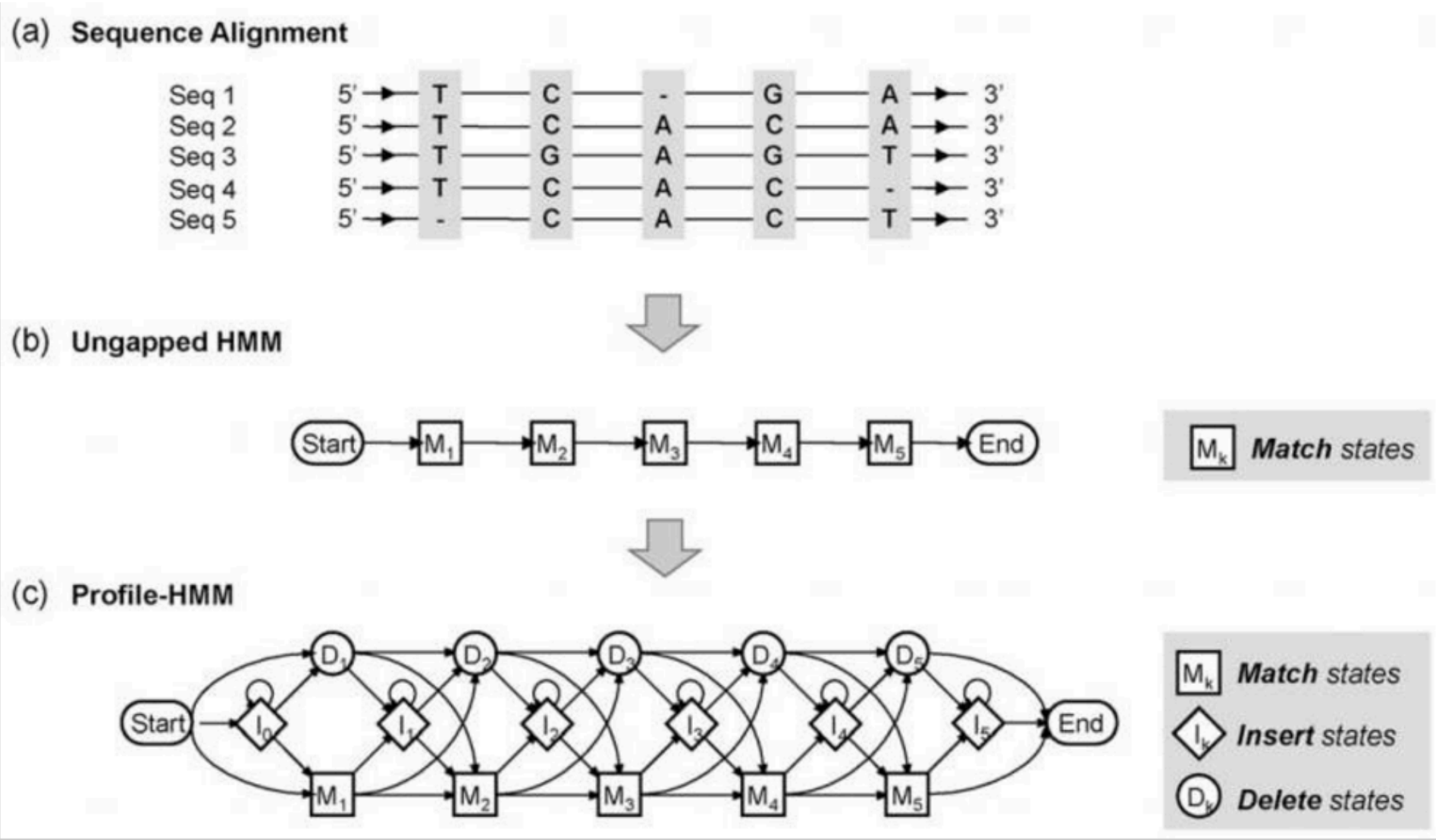
HMM warmup: a model of aligned sequences

```
GNA.GSAIG
SHA.GTCLG
SNA.GTAIG
SHA.GTCLG
GHA.GTCLG
GNA.GSALG
NNA.SITIG
NHA.GVALG
```

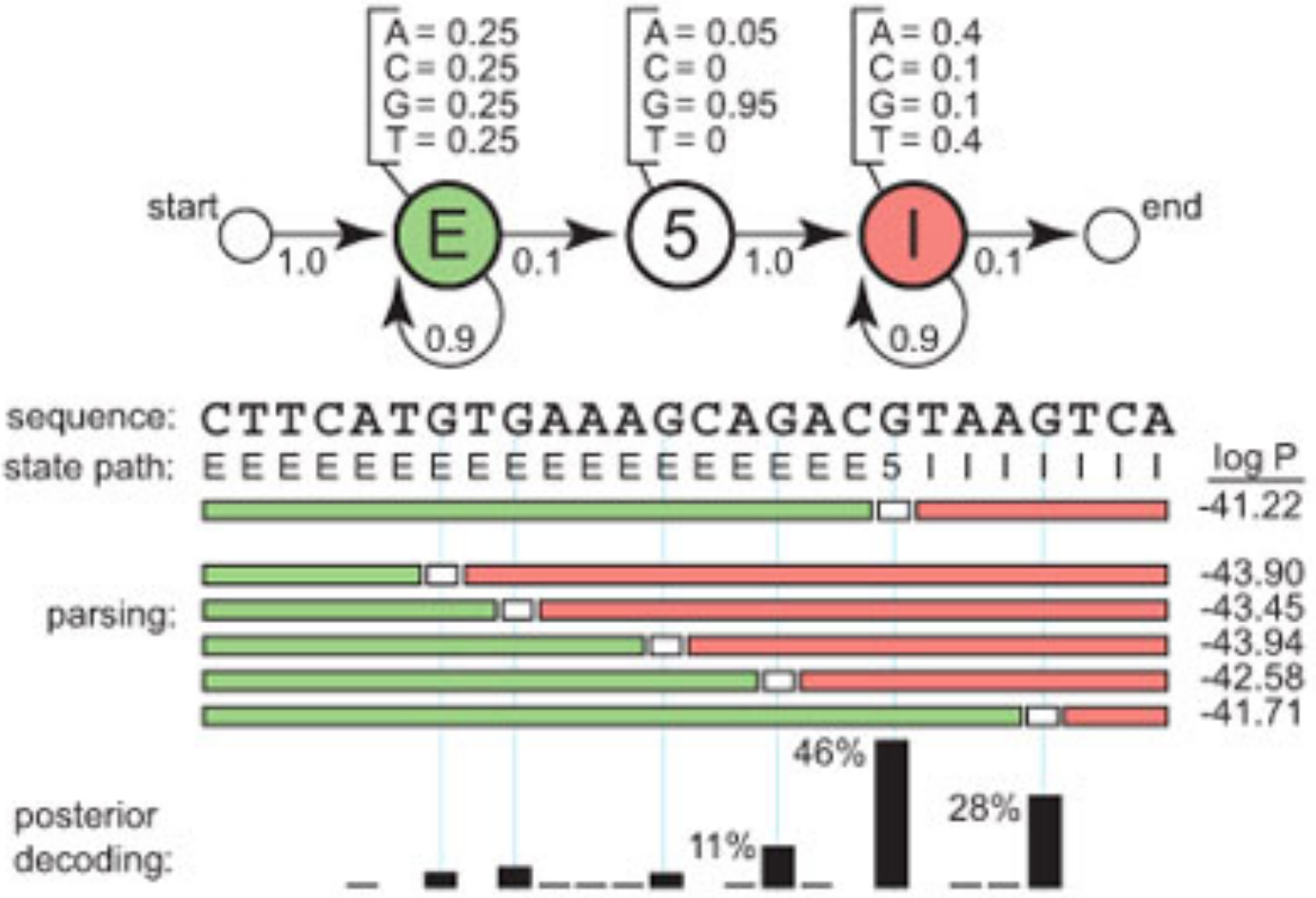


	S1	S2	S3	S4	S5
A	0.01	0.03	0.89	*0.05*	0.01
G	0.3	0.01	0.01	*0.05*	0.82
H	0.01	0.5	0.01	*0.05*	0.01
N	0.2	0.4	0.01	*0.05*	0.01
S	0.3	0.01	0.01	0.05	0.15
…					

Profile HMMs



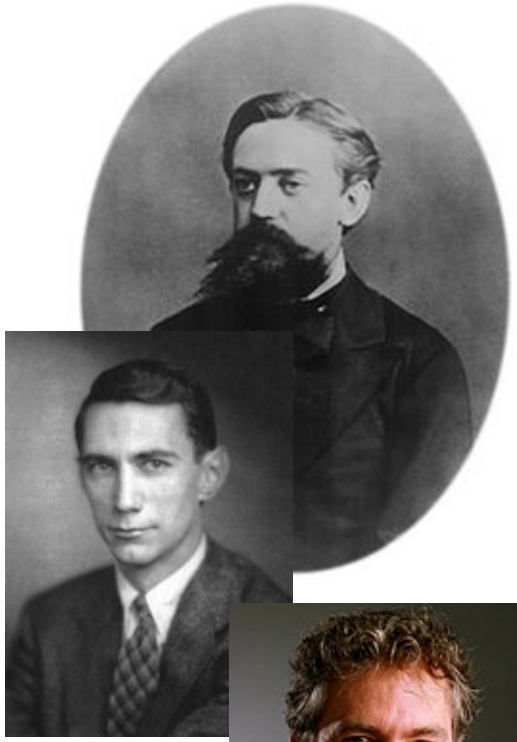
Gene Finding



WHAT IS AN HMM?

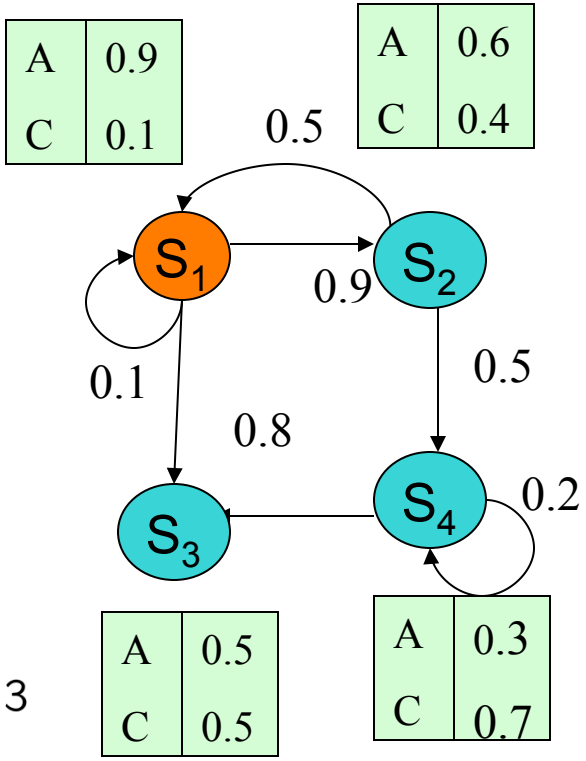
HMMs: History

- Markov chains: Andrey Markov (1906)
  - Random walks and Brownian motion
- Used in Shannon's work on information theory (1948)
- Baum-Welsh learning algorithm: late 60's, early 70's.
  - Used mainly for speech in 60s-70s.
- Late 80's and 90's: David Haussler (major player in learning theory in 80's) began to use HMMs for modeling biological sequences
- Mid-late 1990's: Dayne Freitag/Andrew McCallum
  - Freitag thesis with Tom Mitchell on IE from Web using logic programs, grammar induction, etc.
  - McCallum: multinomial Naïve Bayes for text
  - With McCallum, IE using HMMs on CORA
- …



What is an HMM?

- Generative process:
  - Choose a *start state* S_1 using $Pr(S_1)$
  - For $i=1\ldots n$:
    - *Emit* a *symbol* x_i using $Pr(x|S_i)$
    - *Transition* from S_i to S_j using Pr(S'|S)

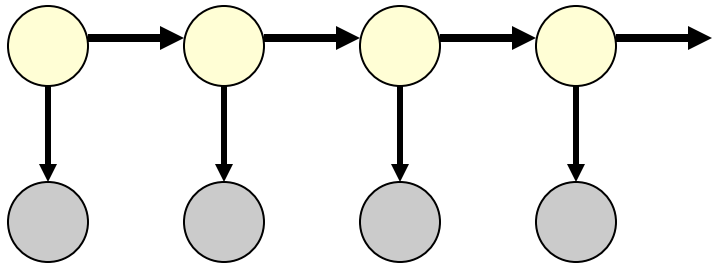


$$\Pr(AACA) = \sum_{ijkl} \Pr(AACA, S_i S_j S_k S_l)$$

$$\Pr(AACA, S_i S_j S_k S_l) = \Pr(S_i)\Pr(A|S_i)\Pr(S_j|S_i)..\Pr(A|S_l)$$

$$\Pr(AACA, S_1 S_2 S_4 S_4) = 1 \times 0.9 \times 0.9 \times 0.6 \times 0.5 \times 0.7 \times 0.2 \times 0.3$$

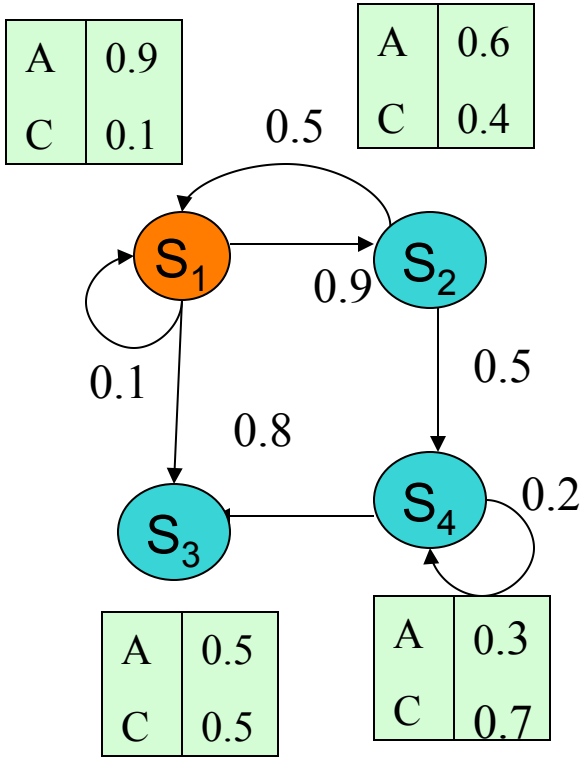
- Usually the token sequence $x_1 x_2 x_3 \ldots$is observed and the state sequence $S_1 S_2 S_3 \ldots$ is not ("hidden")
- An HMM is a special case of a Bayes net



What is an HMM?

- Generative process:
  - Choose a *start state* S_1 using $Pr(S_1)$
  - For $i=1\ldots n$:
    - *Emit* a *symbol* x_i using $Pr(x|S_i)$
    - *Transition* from S_i to S_j using Pr(S'|S)

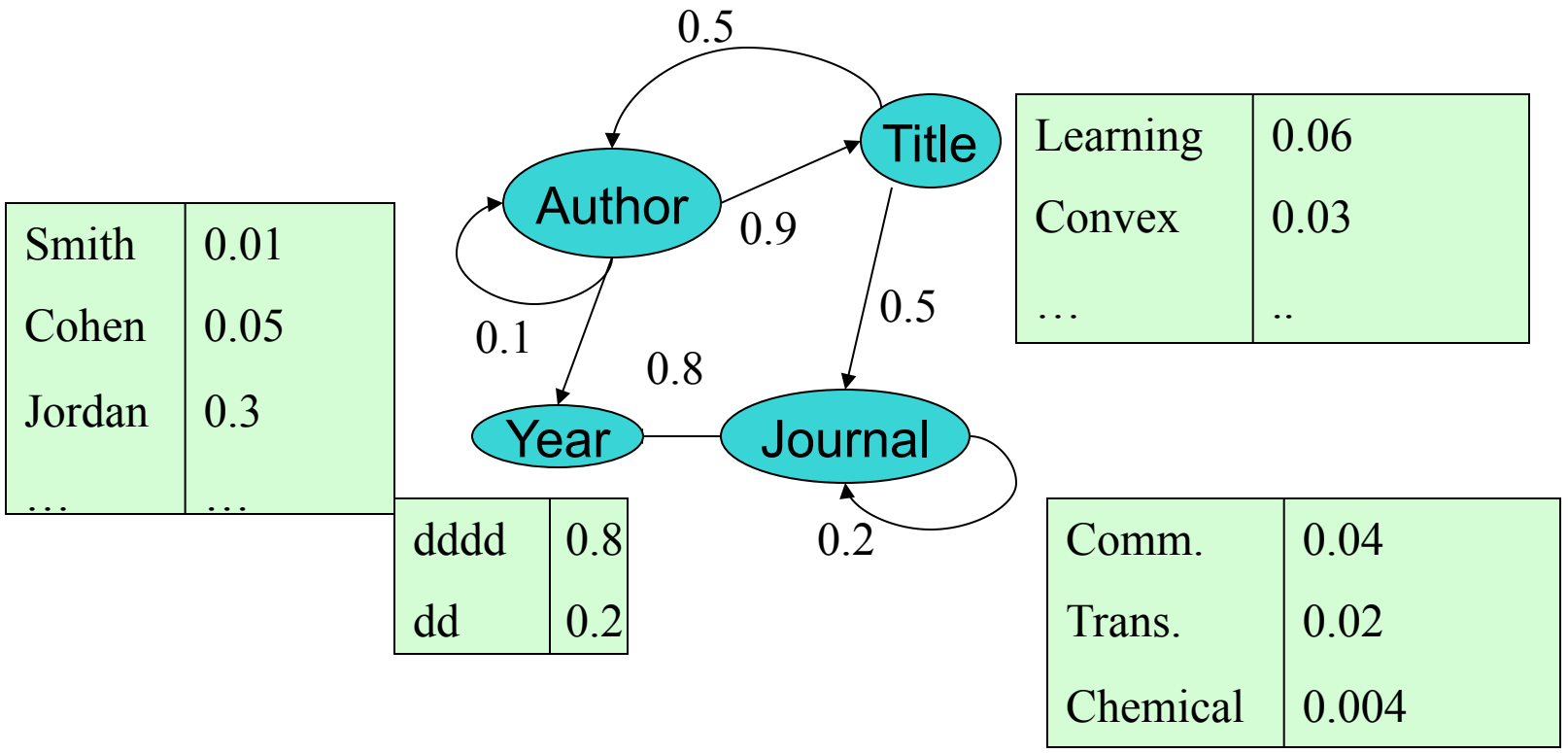
- Some key operations:
  - Given sequence $x_1x_2x_3\ldots$ find the *most probable* hidden state *sequence* $S_1S_2S_3\ldots$
    - We can do this efficiently! **Viterbi**
  - Given sequence $x_1x_2x_3\ldots$ find $Pr(S_j=k|X_1\ldots X_i\ldots=x_1\ldots)$
    - We can do this efficiently! **Forward-Backward**



HMMS FOR NER

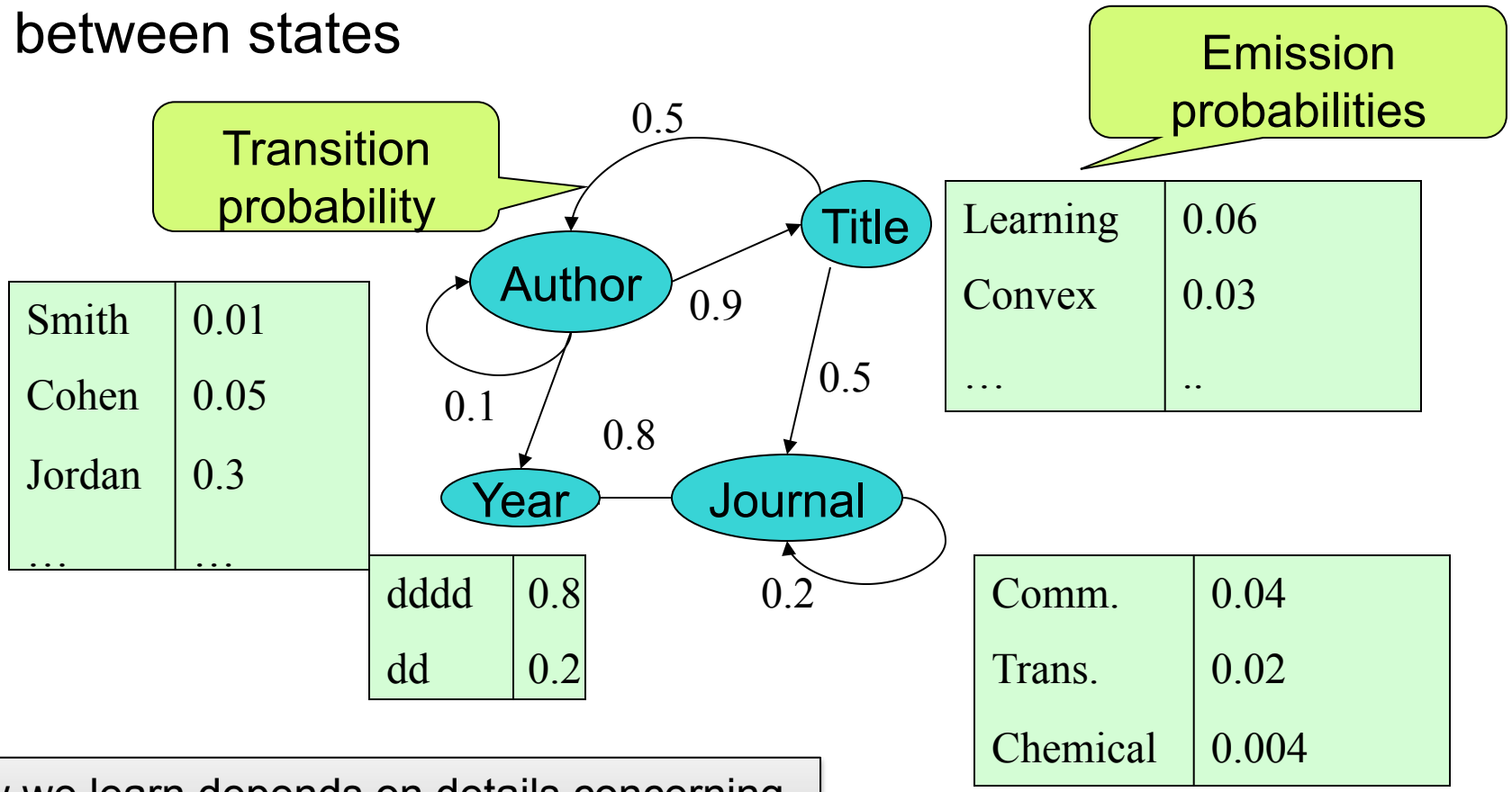
NER with Hidden Markov Models: Learning

- We usually are **given the structure** of the HMM: the vocabulary of states and symbols



NER with Hidden Markov Models: Learning

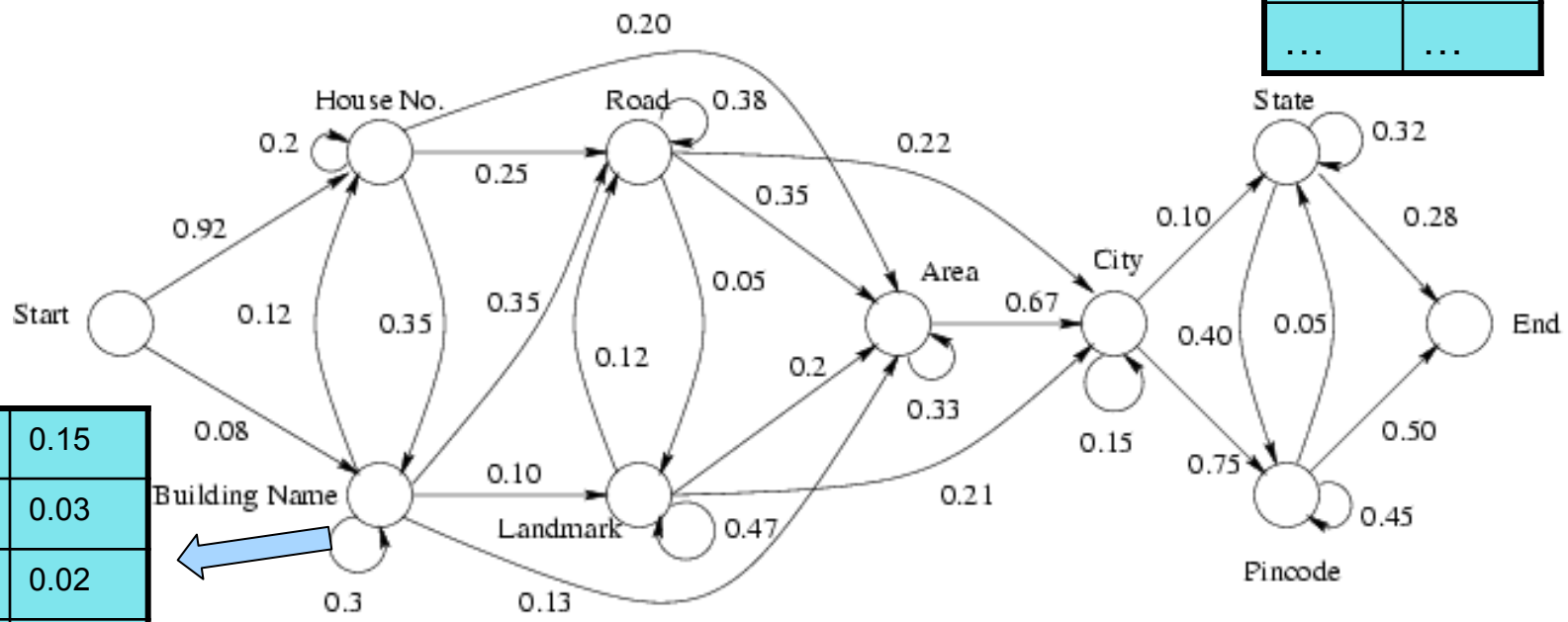
- We **learn** the tables of numbers: *emission probabilities* for each state and *transition probabilities* between states



How we learn depends on details concerning the training data and the HMM structure.

An HMM for Addresses using a "naïve" HMM structure

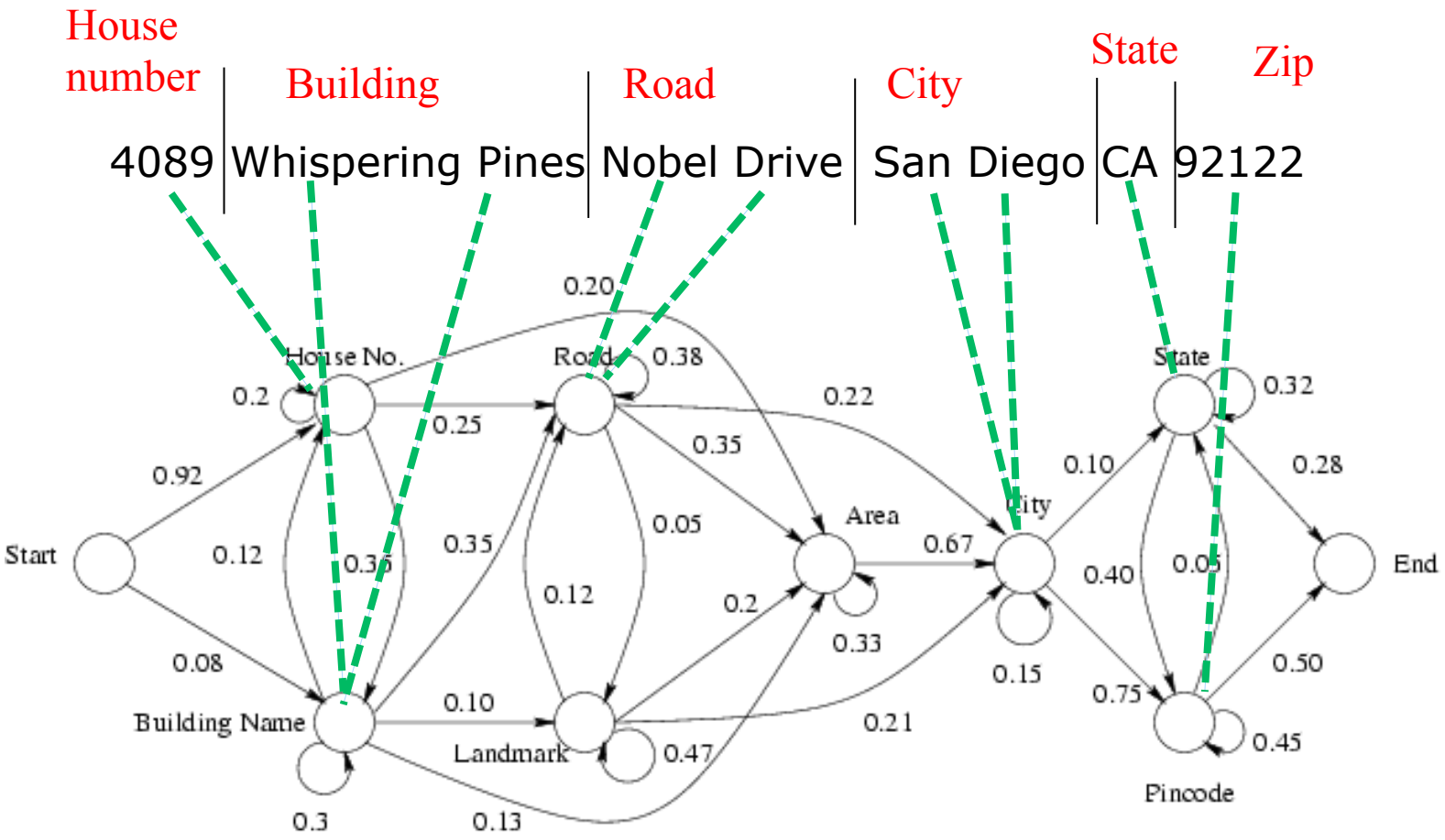
CA	0.15
NY	0.11
PA	0.08
…	…



Hall	0.15
Wean	0.03
Gates	0.02
…	…

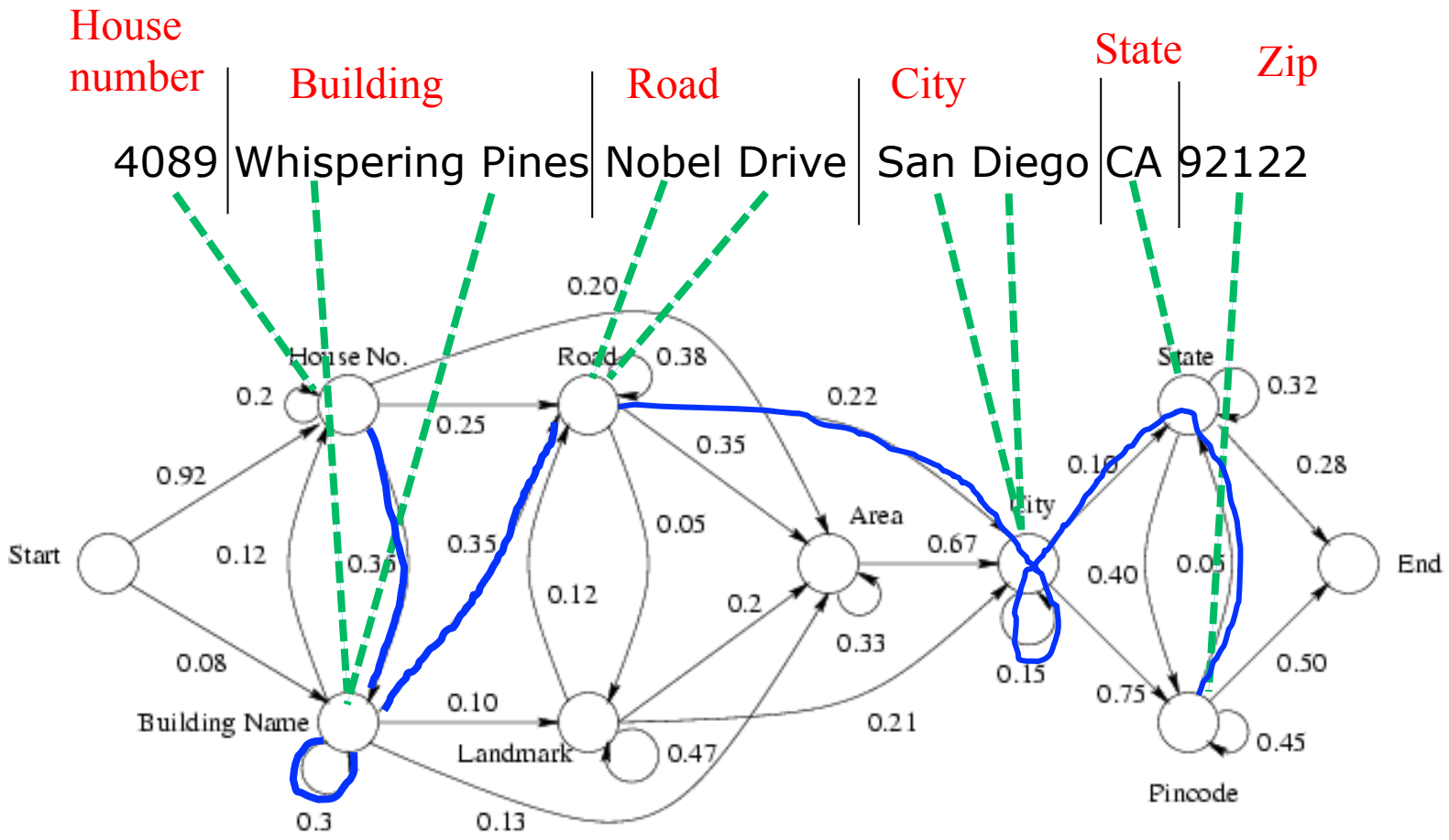
- "Naïve" HMM Structure: One state per entity type, and all transitions are possible

[Pilfered from Sunita Sarawagi, IIT/Bombay]



A key point: with labeled data, we know exactly **which state** emitted **which token.**

This makes it easy to learn the emission probability tables



And: with labeled data, we know exactly **which state transitions** happened.

This makes it easy to learn the transition tables

Breaking it down: Learning parameters for the "naïve" HMM

- Training data defines unique path through HMM!
  - Transition probabilities
    - Probability of transitioning from state i to state j =

$$\frac{\text{number of transitions from i to j}}{\text{total transitions from state i}}$$

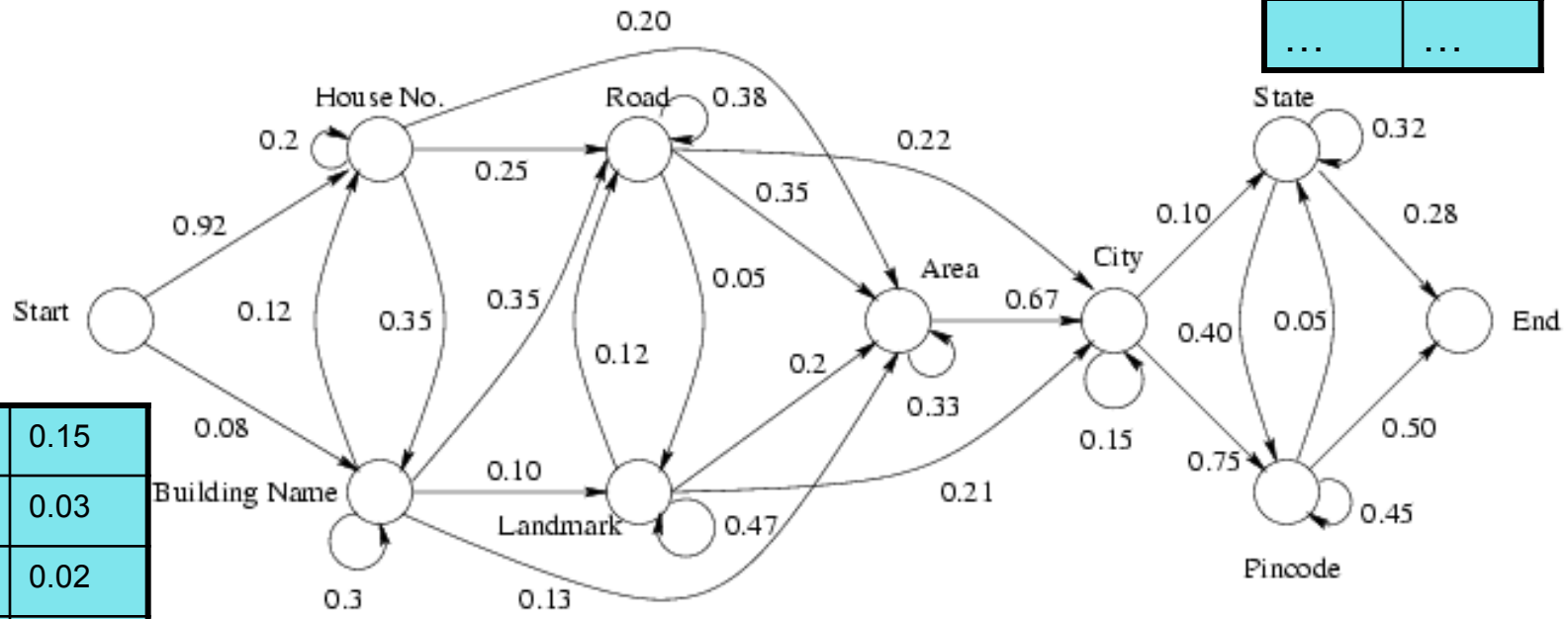
with smoothing, of course

  - Emission probabilities
    - Probability of emitting symbol k from state i =

$$\frac{\text{number of times k generated from i}}{\text{number of transitions from i}}$$

Result of learning: states, transitions, and emissions

CA	0.15
NY	0.11
PA	0.08
…	…



Hall	0.15
Wean	0.03
Gates	0.02
…	…

How do we use this to classify a test sequence?

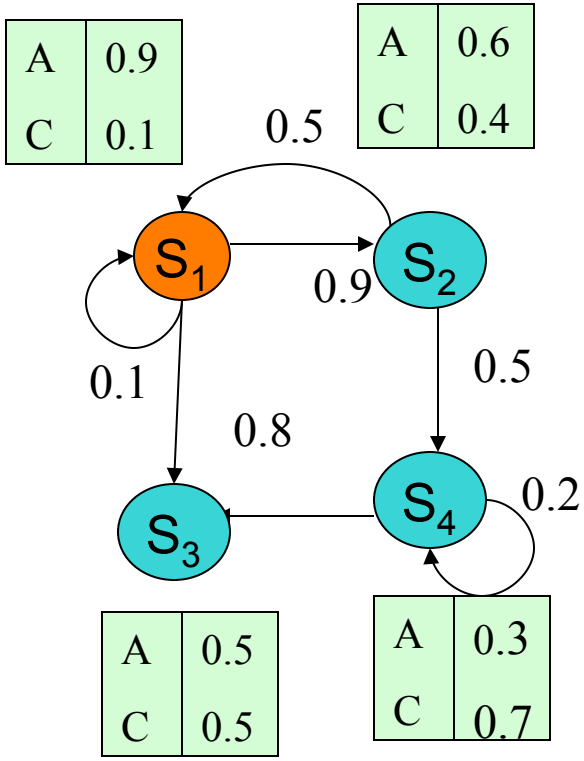
House number	Building	Road	City	State	Zip
4089	Whispering Pines	Nobel Drive	San Diego	CA	92122

What is an HMM?

- Generative process:
  - Choose a *start state* S_1 using $Pr(S_1)$
  - For $i=1\ldots n$:
    - *Emit* a *symbol* x_i using $Pr(x|S_i)$
    - *Transition* from S_i to S_j using Pr(S'|S)

- Some key operations:
  - Given sequence $x_1x_2x_3\ldots$ find the *most probable* hidden state *sequence* $S_1S_2S_3\ldots$
    - We can do this efficiently! **Viterbi**

  - Given sequence $x_1x_2x_3\ldots$ find $Pr(S_j=k|X_1\ldots X_i\ldots=x_1\ldots)$
    - We can do this efficiently! **Forward-Backward**



VITERBI FOR HMMS

Viterbi in pictures

s1	s2	s3	s4	s5	s6
4089	Nobel	Drive	San	Diego	92122

Four states: HouseNum, Road, City, Zip

The slow way: test every possible hidden state sequence <s_1 s_2 … s_6> and see which makes the text most probable (6^4 sequences).

$$\Pr(\text{4089 Nobel Drive San Diego 92122} \mid s_1 s_2 \ldots s_6) =$$

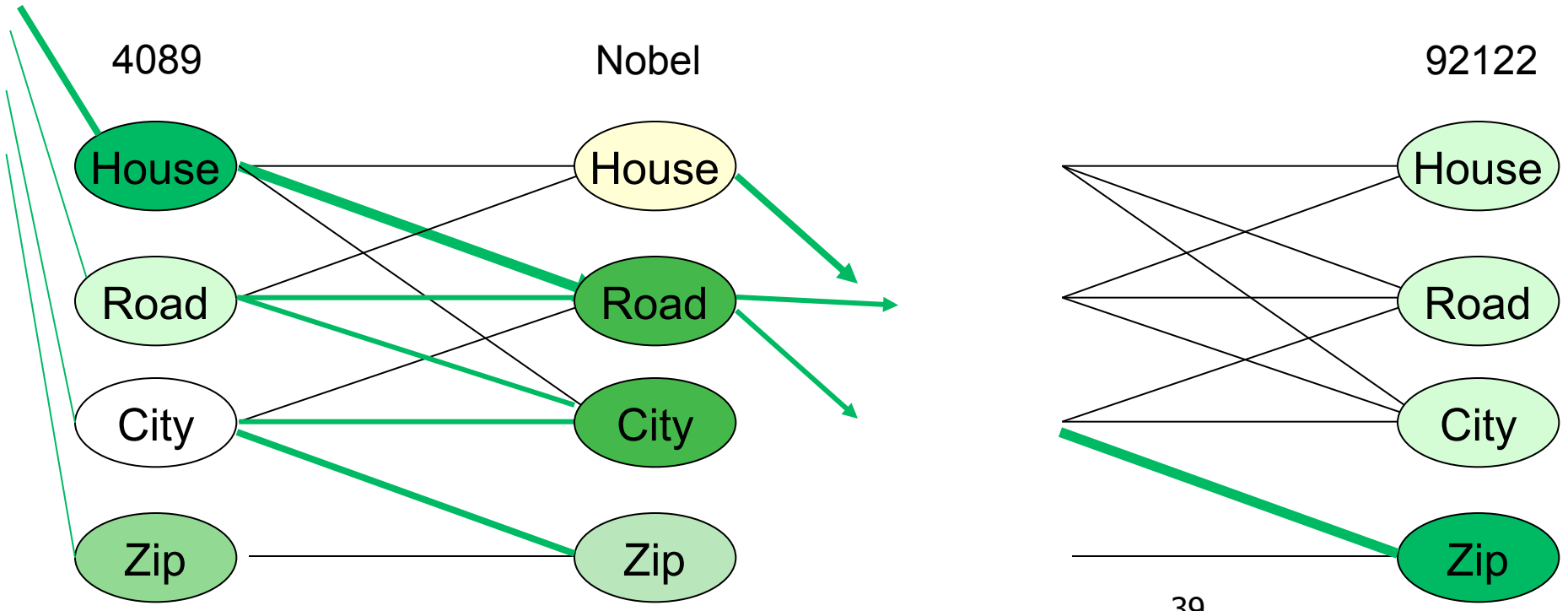
$$\Pr(s_1)\Pr(\text{4089} \mid s_1)\Pr(s_2 \mid s_1)\Pr(\text{Nobel} \mid s_2)\ldots\Pr(s_6 \mid s_5)$$

The fast way: dynamic programming: reduces time from $O(|S|^{|\mathbf{x}|})$ to $O(|\mathbf{x}||S|^2)$

Viterbi in pictures

4089 Nobel Drive San Diego 92122

Circle color indicates Pr(x|s), line width indicates Pr(s'|s)



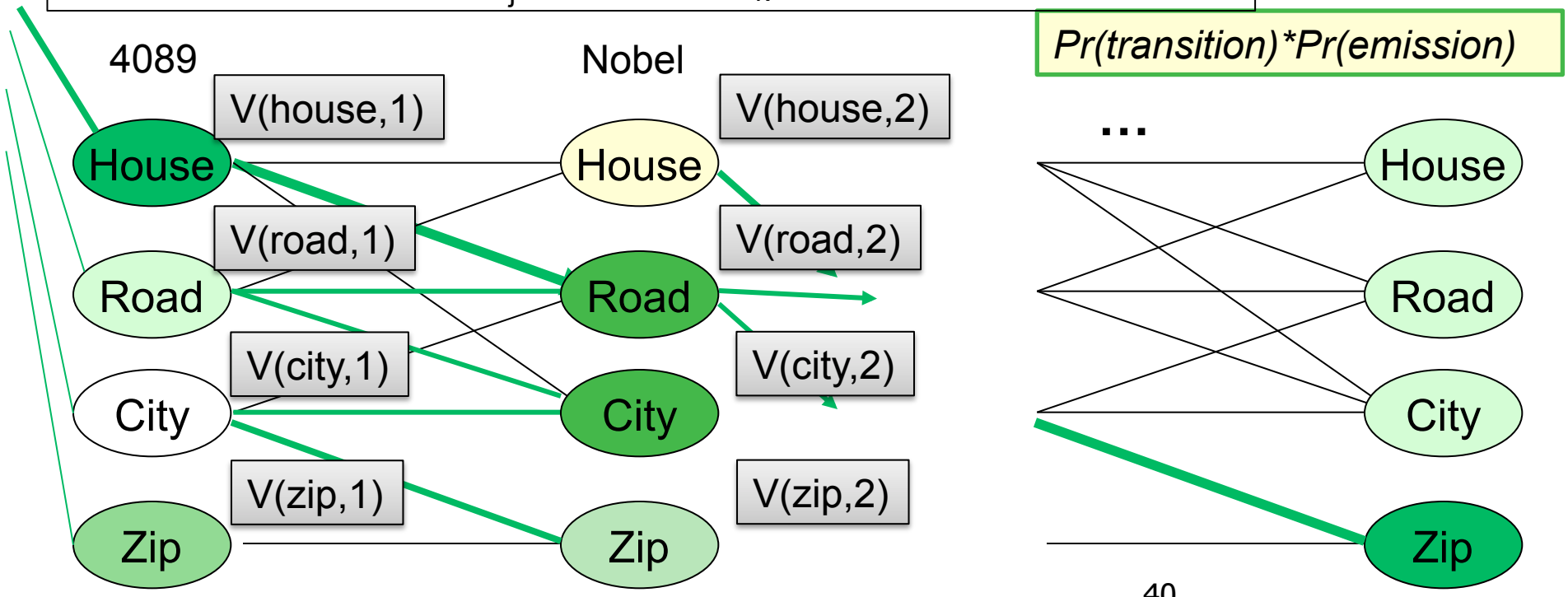
Viterbi algorithm

- Let V be a matrix with |S| rows and |**x**| columns.
- Let ptr be a matrix with |S| rows and |**x**| columns..
  - V(k,j) will be: max over all $s_1 \ldots s_j$: $s_j = k$ of $Prob(x_1 \ldots x_j | s_1 \ldots s_j)$

- For all k: $V(k,1) = Pr(S_1=k) * Pr(x_1|S=k)$
- For j=1,…,|**x**|
  - $V(k,j+1) = Pr(x_j|S=k) * \max_{k'} [Pr(S=k|S'=k') * V(k',j)]$

*Pr(start)*Pr(first emission)*

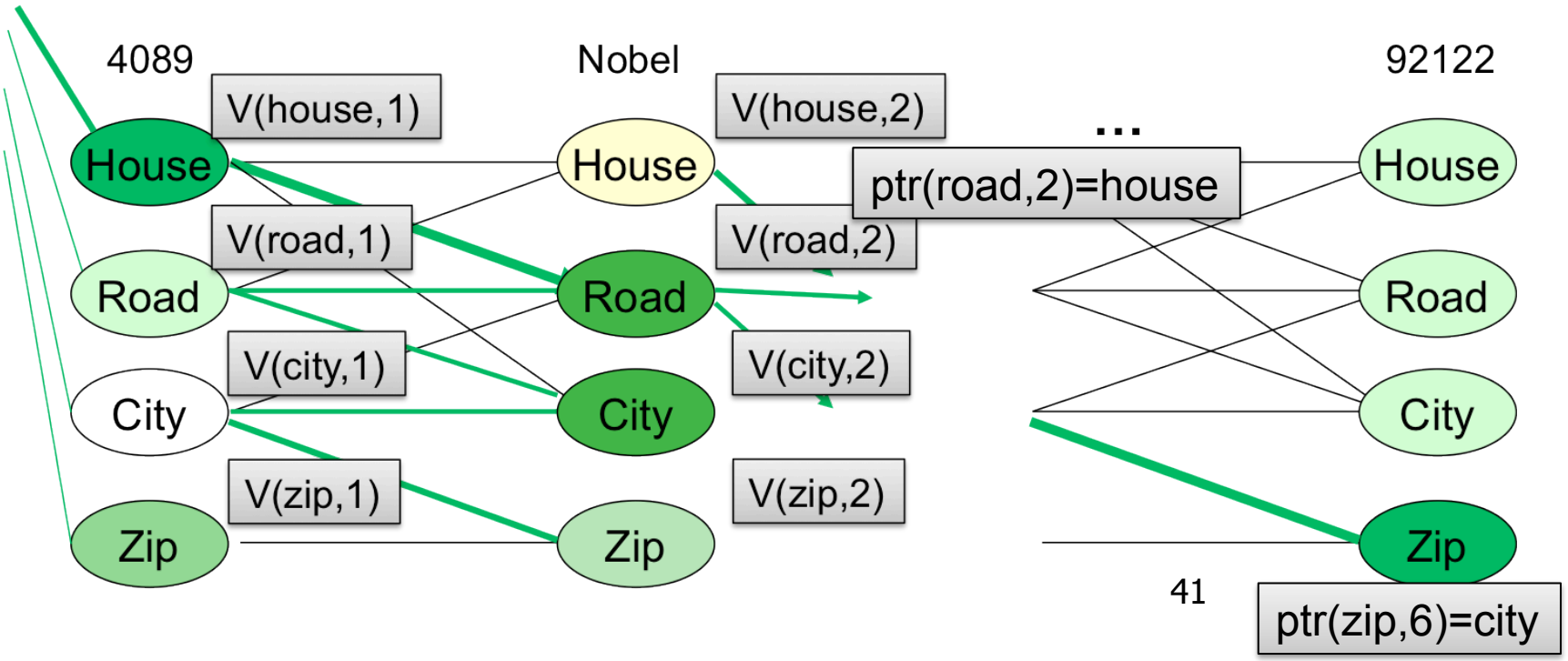
*Pr(transition)*Pr(emission)*



Viterbi algorithm

- Let V be a matrix with |S| rows and |**x**| columns.
- Let ptr be a matrix with |S| rows and |**x**| columns..

- For all k: $V(k,1) = Pr(S_1=k) * Pr(x_1|S=k)$
- For j=1,…,|**x**|
  - $V(k,j+1) = Pr(x_j|S=k) * \max_{k'} Pr(S'=k|S=k') * V(k',j)$
  - $ptr(k,j+1) = \text{argmax}_{k'} Pr(x_j|S=k) * Pr(S=k|S'=k') * V(k',j)$

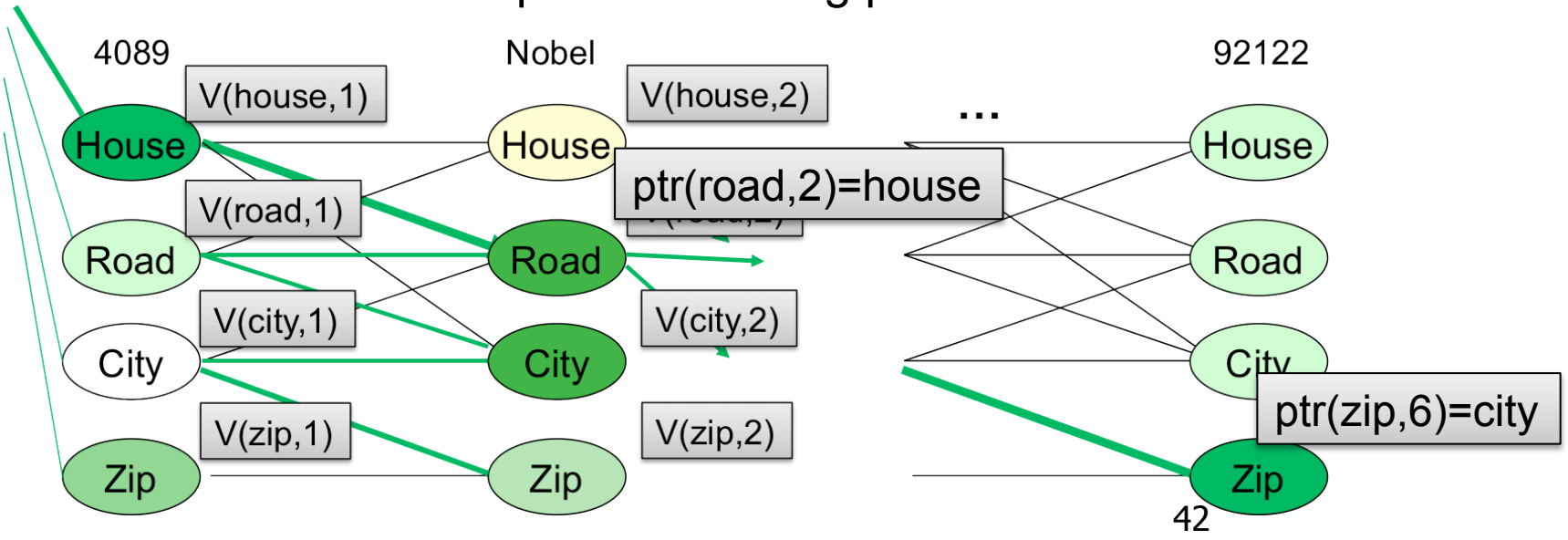


Viterbi algorithm

Implement this in log space with addition instead of multiplication

- Let V be a matrix with |S| rows and |**x**| columns.
- Let ptr be a matrix with |S| rows and |**x**| columns..
- For all k: $V(k,1) = Pr(S_1=k) * Pr(x_1|S=k)$
- For j=1,…,|**x**|-1
  - $V(k,j+1) = Pr(x_j|S=k) * \max_{k'} Pr(S'=k|S=k') * V(k',j)$
  - $ptr(k,j+1) = \text{argmax}_{k'} Pr(x_j|S=k) * Pr(S=k|S'=k') * V(k',j)$

- Let k* = argmax k V(k,|**x**|) -- *the best final path*
- Reconstruct the best path to k* using ptr



Breaking it down: NER using the "naïve" HMM

- Define the HMM structure:
  - one state per entity type
- Training data defines unique path through HMM for each labeled example
  - Use this to estimate transition and emission probabilities
- At test time for a sequence **x**
  - Use Viterbi to find sequence of states **s** that maximizes Pr(**x**|**s**)
  - Use **s** to derive labels for the sequence **x**

What forward-backward computes

Like probabilistic inference: Pr(X|E)

Parsing addresses

House number | Building | Road | City | State | Zip

4089 | Whispering | Pines | Nobel Drive | San Diego | CA | 92122

What is the best prediction for this token?

for this token?

Parsing citations

Author

P.P.Wangikar, T.P. Graycar, D.A. Estell, D.S. Clark, J.S. Dordick (1993) Protein and Solvent Engineering of Subtilising BPN' in Nearly Anhydrous Organic Media J.Amer. Chem. Soc. 115, 12231-12237.

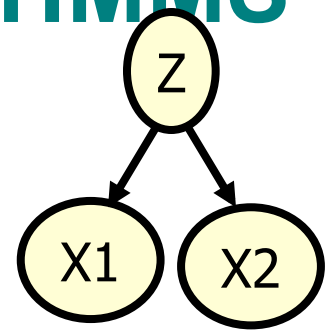
Title

Journal

Volume

THE FORWARD-BACKWARD ALGORITHM FOR HMMS

F-B could also be used to learn HMMS with *hidden* variables



Hidden variables: what if some of your data is not completely observed?

Method (Expectation-Maximization, EM):

1. Estimate parameters somehow or other.
2. Predict unknown values from your estimated parameters (**Expectation step**)
3. Add pseudo-data corresponding to these predictions, *weighting each example by confidence in its correctness.*
4. Re-estimate parameters using the extended dataset (real + pseudo-data).
   - You find the MLE or MAP values of the parameters. (**Maximization step**)
5. Repeat starting at step 2….

Z	X1	X2
ugrad	<20	facebook
ugrad	20s	facebook
grad	20s	thesis
grad	20s	facebook
prof	30+	grants
?	<20	facebook
?	30s	thesis

Possible application of F-B: partially labeled data

McCallum & Culotta, AAAI 2005

1. System proposes a segmentation:

House number	Road	City	State	Zip
4089	Whispering Pines Nobel Drive	San Diego	CA	92122

2. User corrects errors in segmentation:

House number	Building	Road	Didn't really look at this
4089	Whispering Pines	Nobel Drive	San Diego CA 92122

3. Depending on how careful the user is – you might want to use only some of the labeled data

For EM: what is the prediction for this token?

More applications of F-B

User, or some other source, gives a partial segmentation:

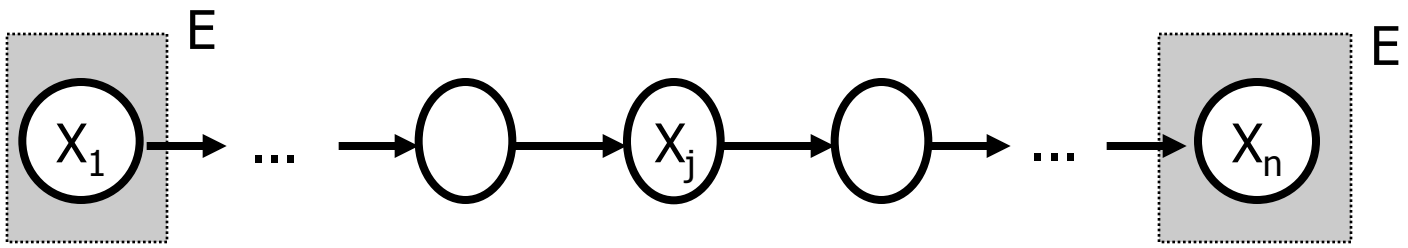
O B I O
Mid-late 1990's : Dayne Freitag and A. McCallum.

O B I O
Freitag thesis with Tom Mitchell on IE from Web using logic programs, grammar induction, etc.
McCallum: multinomial Naïve Bayes for text
With McCallum, IE using HMMs on CORA

…

Forward backward warmup 1
BP on chains

A special case: linear chain networks



$$P(X_1,\ldots,X_n) = P(X_n \mid X_{n-1})P(X_{n-1} \mid X_{n-2})\ldots P(X_2 \mid X_1)P(X_1)$$

$$P(X_j \mid x_1, x_n) = \frac{P(x_n \mid X_j, x_1)P(X_j \mid x_1)}{P(x_n \mid x_1)}$$

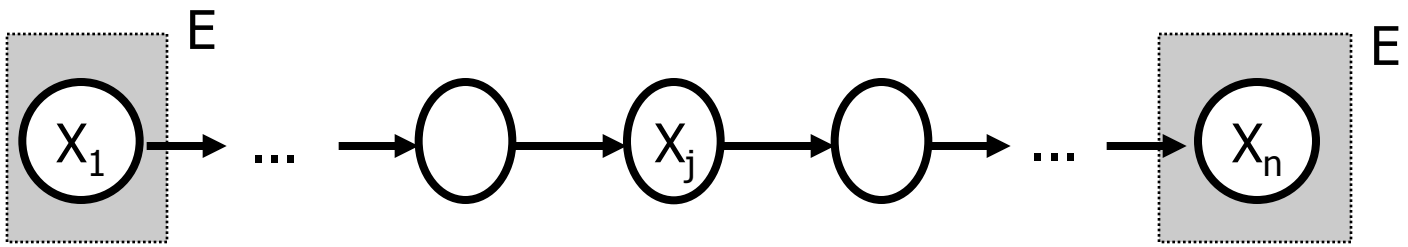
d-separation

$$= c \cdot P(x_n \mid X_j)P(X_j \mid x_1)$$

"backward" (evidential)

"forward" (causal)

A special case: linear chain networks

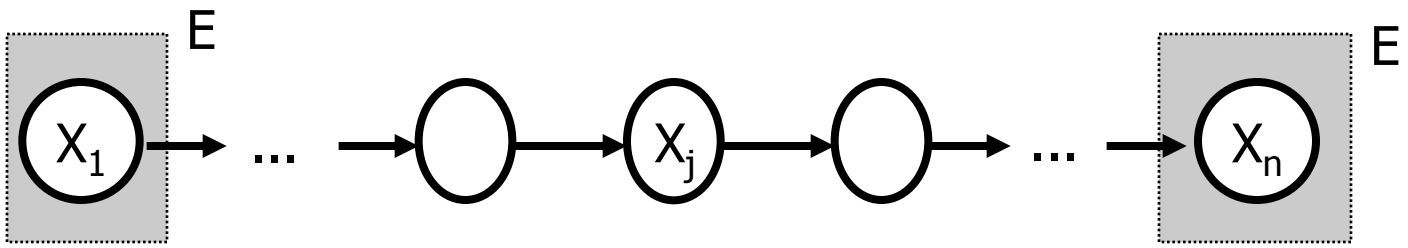


$$P(X_1,\ldots,X_n) = P(X_n \mid X_{n-1})P(X_{n-1} \mid X_{n-2})\ldots P(X_2 \mid X_1)P(X_1)$$

$$P(X_j \mid x_1, x_n) = c \cdot P(x_n \mid X_j)P(X_j \mid x_1)$$

Fwd: $$P(X_j = x \mid x_1) = \sum_{x'} \underbrace{P(X_{j-1} = x' \mid x_1)}_{\text{Recursion! (fwd)}} \underbrace{P(X_j = x \mid X_{j-1} = x')}_{\text{CPT entry}}$$

A special case: linear chain networks



$$P(X_1,\ldots,X_n) = P(X_n \mid X_{n-1})P(X_{n-1} \mid X_{n-2})\ldots P(X_2 \mid X_1)P(X_1)$$

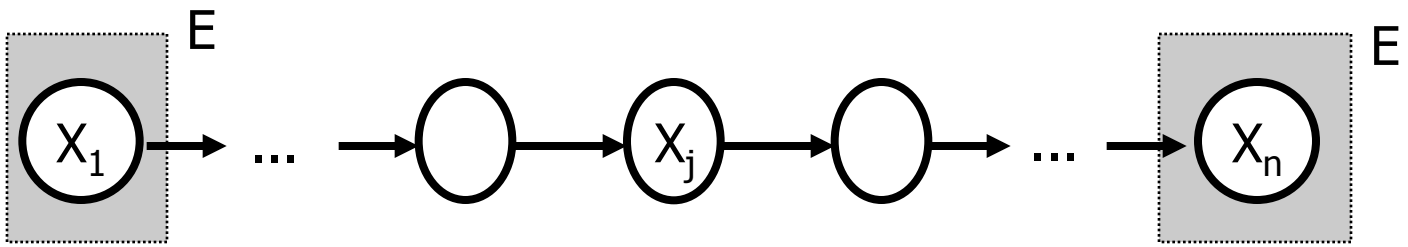
$$P(X_j \mid x_1, x_n) = \alpha \cdot P(x_n \mid X_j)P(X_j \mid x_1)$$

Back: $$P(x_n \mid X_j) = \sum_{x'} P(x_n, X_{j+1} = x' \mid X_j)$$

$$= \sum_{x'} \underbrace{P(x_n \mid X_{j+1} = x', X_j)}_{\text{Recursion backward}} \underbrace{P(X_{j+1} = x' \mid X_j)}_{\text{CPT}}$$

Chain rule

A special case: linear chain networks



$$P(X_1,\ldots,X_n) = P(X_n \mid X_{n-1})P(X_{n-1} \mid X_{n-2})\ldots P(X_2 \mid X_1)P(X_1)$$

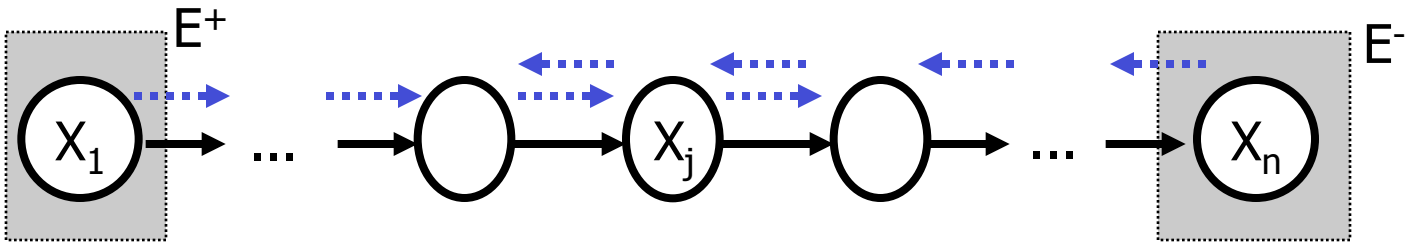
$$P(X_j \mid x_1, x_n) = \alpha \cdot \underbrace{P(x_n \mid X_j)}_{\text{“backward”}} \underbrace{P(X_j \mid x_1)}_{\text{“forward”}}$$

Instead of recursion:

- iteratively compute $P(X_j|x_1)$ from $P(X_{j-1}|x_1)$ – the forward probabilities
- iteratively compute $P(x_n|X_j)$ from $P(x_n|X_{j+1})$ – the backward probabilities
- can view the forward computations as passing a “message” forward
  - and vice versa

Linear-chain message passing

$P(X_j|E) = P(X|E+)P(X|E-)$ … true by d-separation



Pass <u>forward</u>: $P(X_j|E^+)$…computed from $P(X_{j-1}|E^+)$ and CPT for X_j

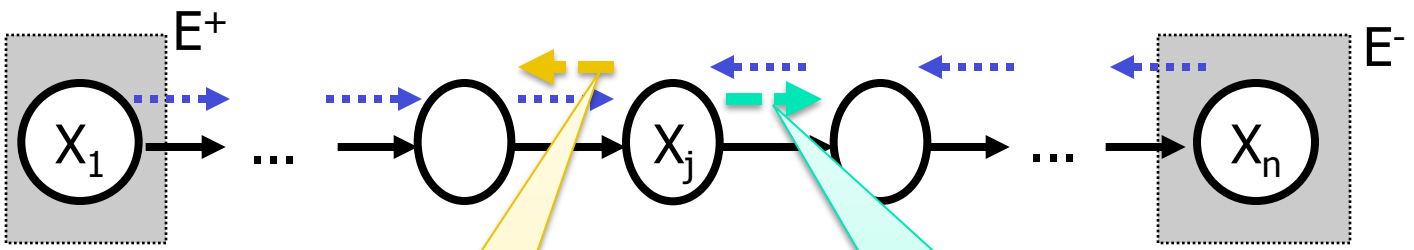
$$P(X_j = x \mid x_1) = \sum_{x'} P(X_{j-1} = x' \mid x_1) P(X_j = x \mid X_{j-1} = x')$$

Pass <u>backward</u>: $P(X_j|E^-)$…computed from $P(X_{j+1}|E^-)$ and CPT for X_{j+1}

$$P(x_n \mid X_j = x) = \sum_{x'} P(x_n \mid X_{j+1} = x') P(X_{j+1} = x' \mid X_j = x)$$

Linear-chain message passing

$P(X_j|E) = P(X|E+)P(X|E-)$ … true by d-separation



$Pr(E-|X_j=b\_person)=0.15$
$Pr(E-|X_j=i\_person)=0.731$
…

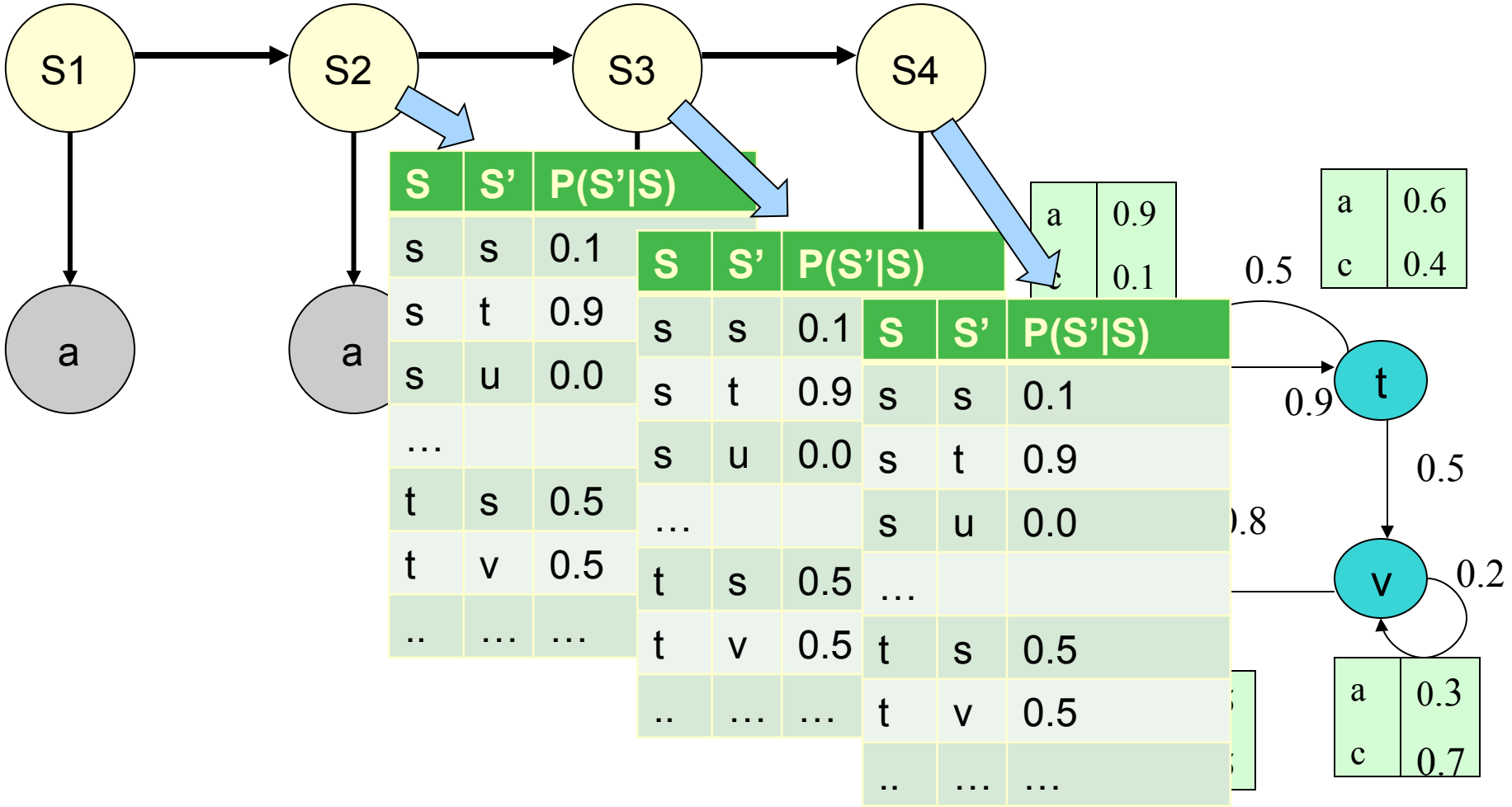
$Pr(X_j=b\_person|E+)=0.2$
$Pr(X_j=i\_person|E+)=0.36$
…

Forward backward warmup 2 simple “dynamic” Bayes nets

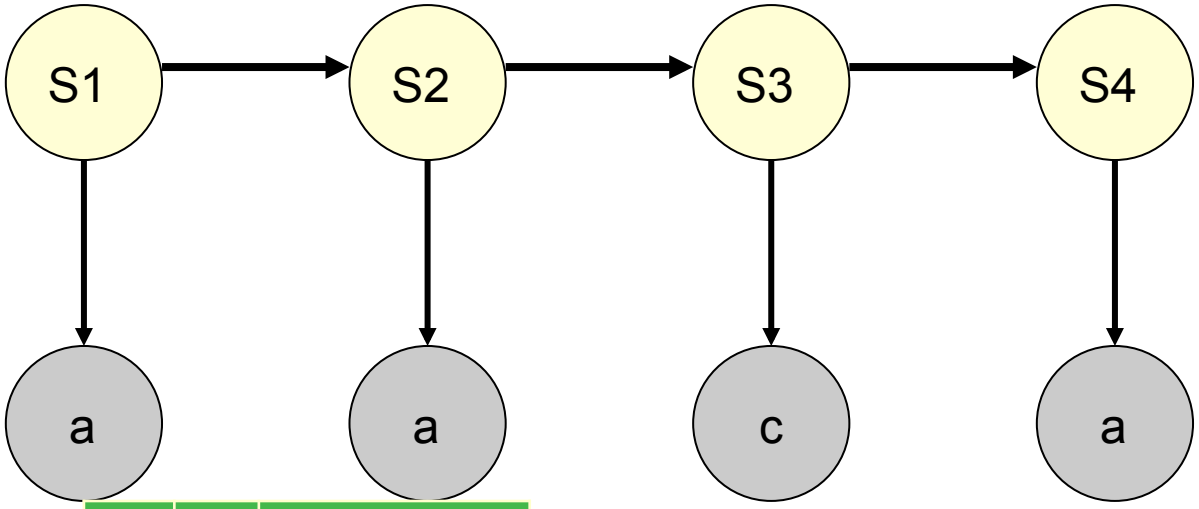
An HMM-like Bayes Net

S	P(S)
s	1.0
t	0.0
u	0.0
v	0.0

"Tied" parameters



An HMM-like Bayes Net

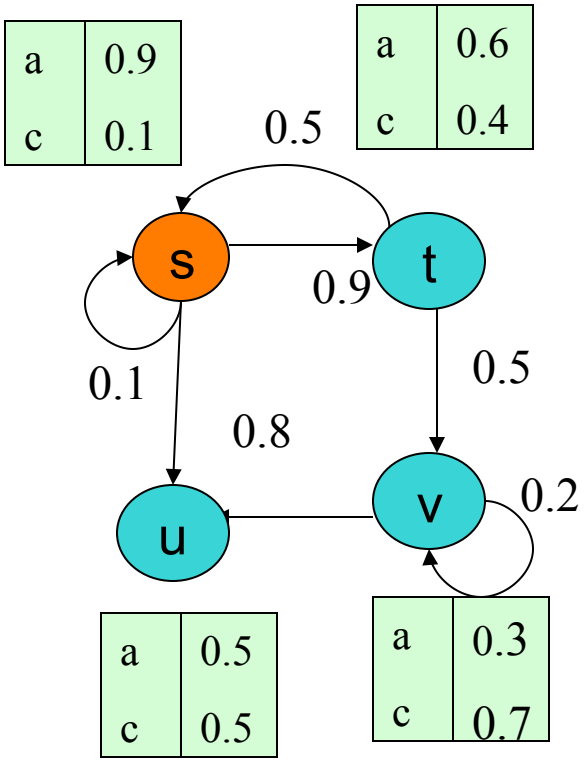


S	X	P(X
s	a	0.9
s	c	0.1
t	a	0.6
t	c	
..	…	…

S	X	P(X|S)
s	a	0.9
s	c	0.1
t	a	0.6
t	c	0.4
..	…	…

…

"Tied" parameters



Forward-backward for a small example

	S1	S2	S3	S4	S5	S6
x	4089	Nobel	Drive	San	Diego	92122

Four states: HouseNum, Road, City, Zip

What is Pr(S3=road|evidence)?

The slow way: use the joint.
There are $4^6 = 4096$ entries here

Is there a faster way? Use BP – the special case for this problem for HMMs is called forward-backward

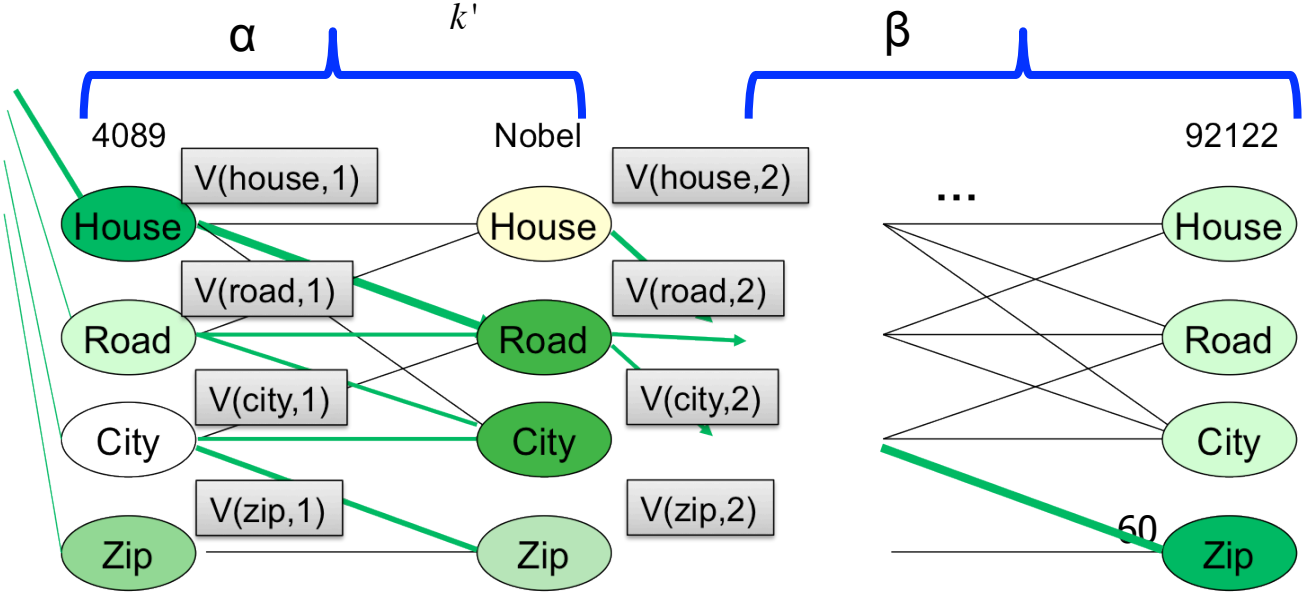
Forward Backward

- Let α and β be matrices with |S| rows and |**x**| columns.
  - α is "forward probability", β is "backward probability"
- $\alpha(k,1) = V(k,1) = \Pr(S_1=k) * \Pr(x_1|S=k)$
- $\beta(k,|\mathbf{x}|+1) = 1$
- For j=1 to |**x**|-1:

$$\alpha(k, j+1) = \Pr(x_{j+1} \mid S_{j+1} = k) \sum_{k'} \Pr(S = k \mid S' = k')\alpha(k', j)$$

- For j = |**x**| down to 2:

$$\beta(k, j-1) = \sum_{k'} \Pr(x_j \mid S = k')\Pr(S = k \mid S = k')\beta(k', j)$$



Forward Backward

- Let α and β be matrices with |S| rows and |**x**| columns.
  - α is "forward probability", β is "backward probability"
- α(k,1) = V(k,1) = Pr(S_1=k) * Pr(x_1|S=k)
- β(k,|**x**|+1) = 1
- For j=1 to |**x**|-1:

$$\alpha(k, j+1) = \Pr(x_{j+1} \mid S_{j+1} = k) \sum_{k'} \Pr(S = k \mid S' = k') \alpha(k', j)$$

- For j = |**x**| down to 2:

$$\beta(k, j-1) = \sum_{k'} \Pr(x_j \mid S = k') \Pr(S = k \mid S = k') \beta(k', j)$$

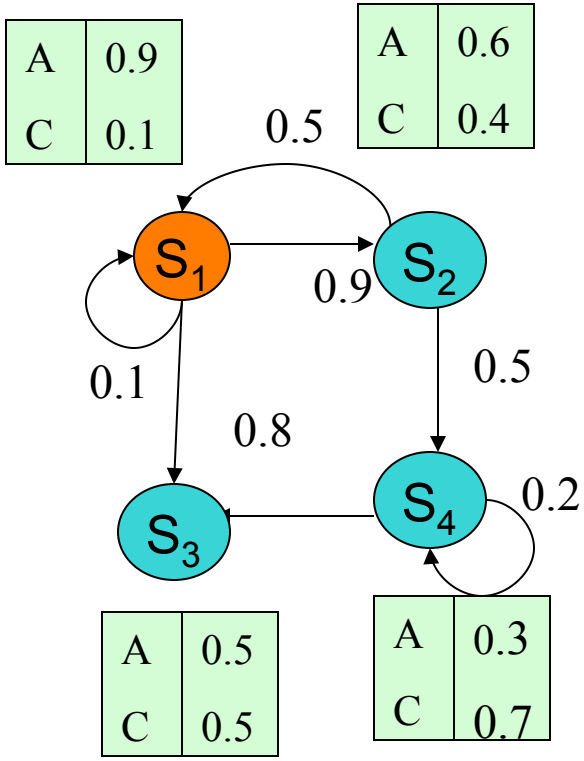
- Now we can compute expectations over the hidden variables:

$$\Pr(S_j = k \mid \mathbf{x}) = \alpha(k, j) \beta(k, j)$$

- …which lets us classify tokens, or use EM to learn (for HMMs, it's called Baum-Welch)

What is an HMM?

- Generative process:
  - Choose a *start state* S_1 using $Pr(S_1)$
  - For $i=1\ldots n$:
    - *Emit* a *symbol* x_i using $Pr(x|S_i)$
    - *Transition* from S_i to S_j using Pr(S'|S)



- Some key operations:
  - Given sequence $x_1x_2x_3\ldots$ find the *most probable* hidden state *sequence* $S_1S_2S_3\ldots$
    - We can do this efficiently! **Viterbi**
  - Given sequence $x_1x_2x_3\ldots$ find $Pr(S_j=k|X_1\ldots X_i\ldots=x_1\ldots)$
    - We can do this efficiently! **Forward-Backward**

Viterbi in pictures

s1	s2	s3	s4	s5	s6
4089	Nobel	Drive	San	Diego	92122

Four states: HouseNum, Road, City, Zip

Inference task: find $\text{argmax}_{\mathbf{s}} \Pr(\mathbf{s}|\mathbf{x})$

$\text{argmax}_{s1,\ldots,s6} \Pr(S1=s1,\ldots,S6=s6|X1=4089,\ldots,X5=Diego,S6=92122$

The slow way: test every possible hidden state sequence $\langle s_1\ s_2\ \ldots\ s_6\rangle$ and see which makes the text most probable (6^4 sequences).

The fast way: variant* of BP – special case for HMMs is called Viterbi. (By extension we sometimes use "Viterbi" for the analogous operation for DGMs.)

Algorithm left as an exercise for the student.

CONDITIONAL RANDOM FIELDS

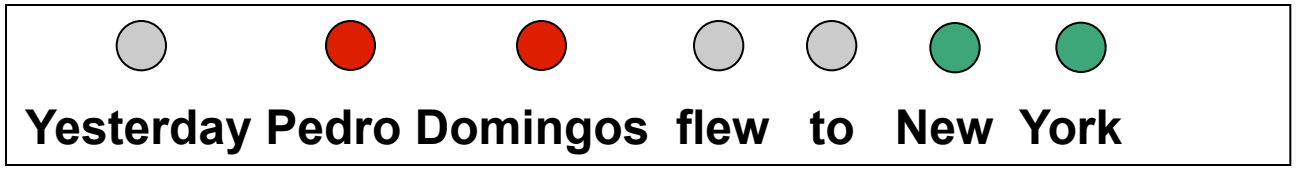
Most common approach: NER by classifying tokens

Given a sentence:

Yesterday Pedro Domingos flew to New

Feature	Value
isCapitalized	yes
numLetters	8
suffix2	-os
word-1-to-right	flew
word-2-to-right	to
…	
thisWord	Domingos

1) Break the sentence into *tokens*, and ***classify*** each token with a label indicating *what sort of entity* it is part of:

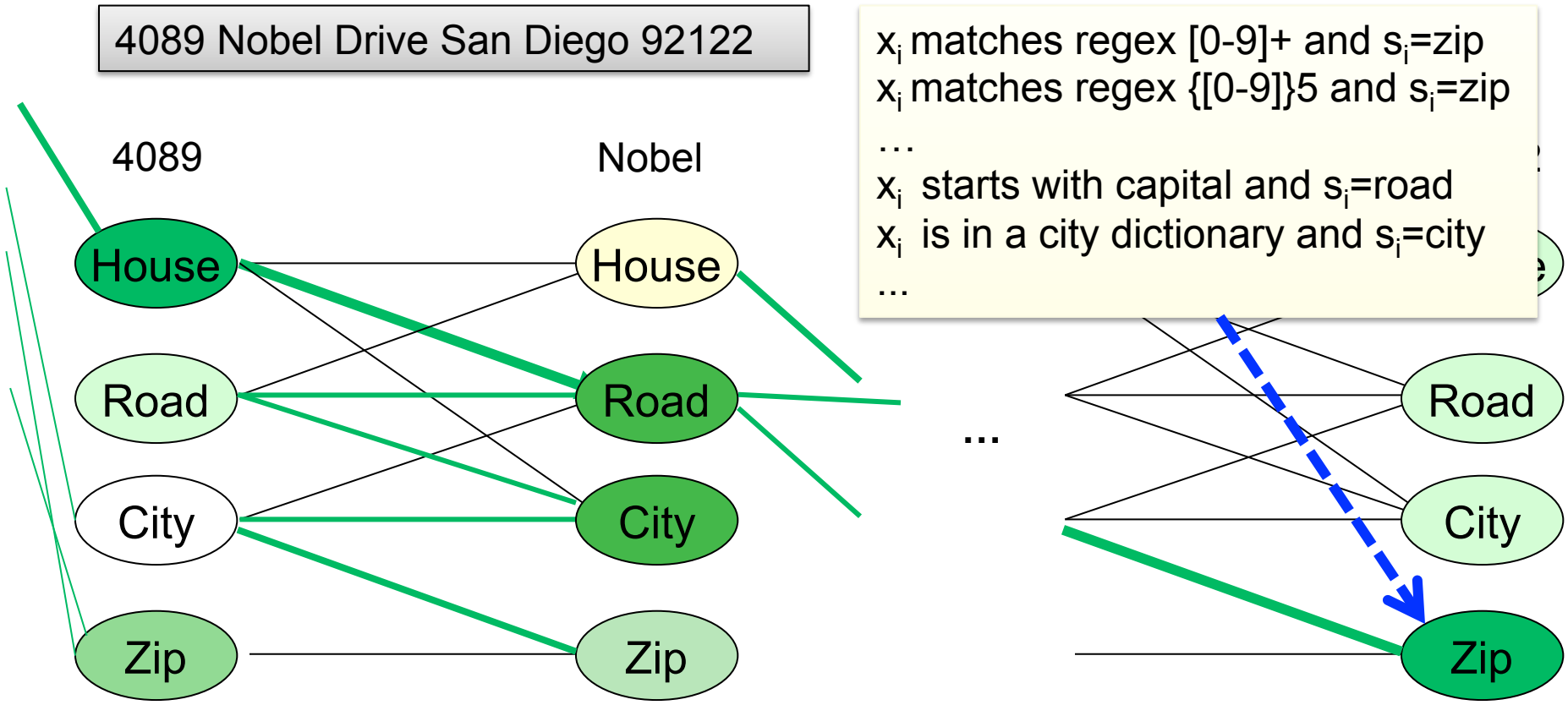


2) Identify names based on the entity labels

Person name: Pedro Domingos
Location name: New York

3) To learn an NER system, use YFCL and **whatever features you want….**

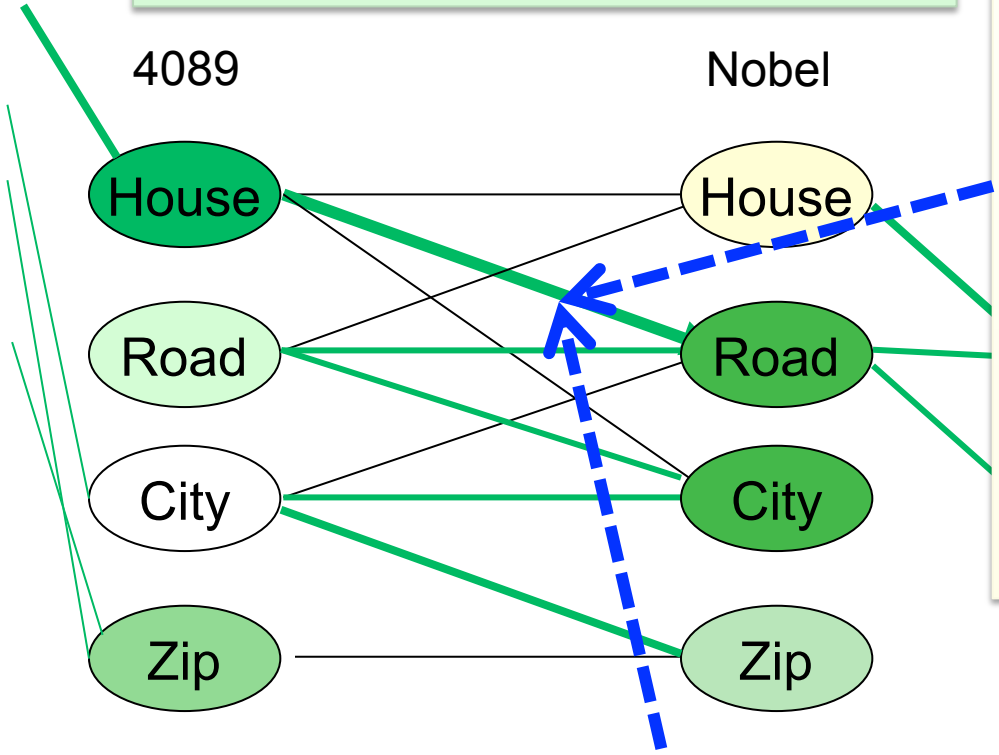
Back to pictures…..



Can we featurize the *way* that the edges and nodes are *weighted*?

Back to pictures…..

we don't really need **anything but edge features**, because an edge feature could ignore part of the edge



x_{i-1} is a digit and x_{ii} is a capitalized word and y_{i-1} = house and y_i=road (f1)

y_{i-1} = house and y_i=road (f2)

…

this is the first transition and y_{i-1} = house and y_i=road (f37)

…

Zip

weight(Y_{i-1}=house , Y_i =road) = 0.23*f1 - 0.61*f2 + ….

A possible learning algorithm….

- Initialize feature weight vector **λ**
- For each labeled example:
  - use **λ** to compute edge weights, node weights for the forward-backward graph
  - use this "machine" to label **x** with **y**
    - e.g. using forward-backward, or something similar
  - if it gets the wrong answer, tweak **λ** to improve performance somehow
    - e.g. using gradient descent

The math: Multiclass logistic regression

$$\Pr(x,y) = \exp(\sum_i \lambda_i f_i(x,y))$$

$$\Pr(y \mid x) = \frac{\exp(\sum_i \lambda_i f_i(x,y))}{\sum_{y'} \exp(\sum_i \lambda_i f_i(x,y'))} = \frac{\exp(\sum_i \lambda_i f_i(x,y))}{Z_\lambda(x)}$$

It's easy to compute this.

Gradient Descent for Logistic Regression

Old notation:

- In batch gradient descent, average the gradient over all the examples $D=\{(x_1,y_1)\ldots,(x_n, y_n)\}$

$$\frac{\partial}{\partial w^j} \log P(D|\mathbf{w}) = \frac{1}{n}\sum_i (y_i - p_i)x_i^j =$$

$$= \frac{1}{n}\sum_{i:x_i^j=1} y_i - \frac{1}{n}\sum_{i:x_i^j=1} p_i$$

Multiclass Logistic Regression

New notation: $D = (x_1, y_1), ..., (x_k, y_k), ...$

$$\frac{\partial}{\partial w^j} \log P(D|\mathbf{w}) = \frac{1}{n} \sum_i (y_i - p_i) x_i^j$$

$$\frac{\partial}{\partial \lambda_i} \Pr(D \mid \vec{\lambda}) = \sum_t \Big(\delta[y_t = k] - p_\lambda(y_t = k \mid x_t) \Big) f_i(x_t, y_t)$$

$$= \sum_t f_i(x_t, y_t) - E_{p_\lambda(Y|x_t)} f_i(Y, x_t)$$

From logistic regression to CRFs

j is over positions in the sequence

Sha and Pereira, 2002

$$P(\vec{y} \mid \vec{x}) = \frac{\prod_j \exp(\sum_i \lambda_i f_i(x_j, y_j, y_{j-1}))}{Z_\lambda(\vec{x})}$$

$$\Pr(y \mid x) = \frac{\exp(\sum_i \lambda_i f_i(x, y))}{Z_\lambda(x)}$$

$$= \frac{\exp(\sum_i \lambda_i F_i(\vec{x}, \vec{y}))}{Z_\lambda(\vec{x})}, \text{ where } F_i(\vec{x}, \vec{y}) = \sum_j f_i(x_j, y_j, y_{j-1})$$

Compute with forward-backward ideas

$$\frac{\partial}{\partial \lambda_i} \Pr(D \mid \vec{\lambda}) = \sum_t f_i(x_t, y_t) - E_{p_\lambda(Y \mid x_t)} f_i(Y, x_t)$$

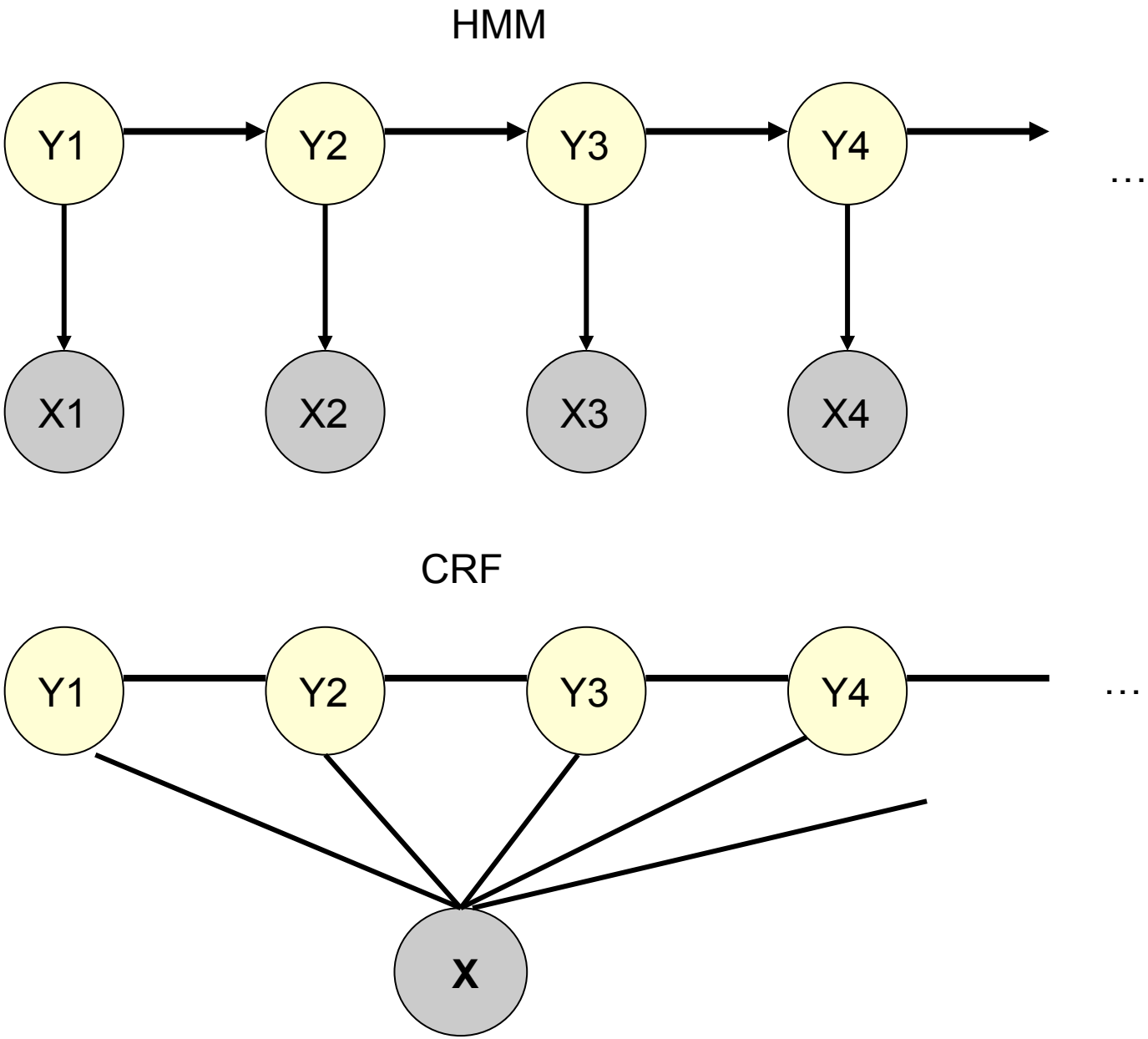
$$\frac{\partial}{\partial \lambda_i} \Pr(D \mid \vec{\lambda}) = \sum_t F_i(\vec{x}_t, \vec{y}_t) - E_{p_\lambda(\vec{Y} \mid \vec{x}_t)} F_i(\vec{Y}, x_t)$$

Conditional random fields

$$F_i(\vec{x},\vec{y}) = \sum_j f_i(x_j, y_j, y_{j-1})$$

$$P(\vec{y} \mid \vec{x}) = \frac{\exp(\sum_i \lambda_i F_i(\vec{x},\vec{y}))}{Z_\lambda(\vec{x})} = \frac{\exp(\sum_i \lambda_i \sum_j f_i(x_j, y_j, y_{j-1})}{Z_\lambda(\vec{x})}$$

- Standard CRF learning method:
  - optimize **λ** to maximize (regularized) conditional log likelihood of the data under this model
  - computing gradient is done by running forward-backward on the data
    - instead of using the weights to "predict" each example's score, as in logistic regression

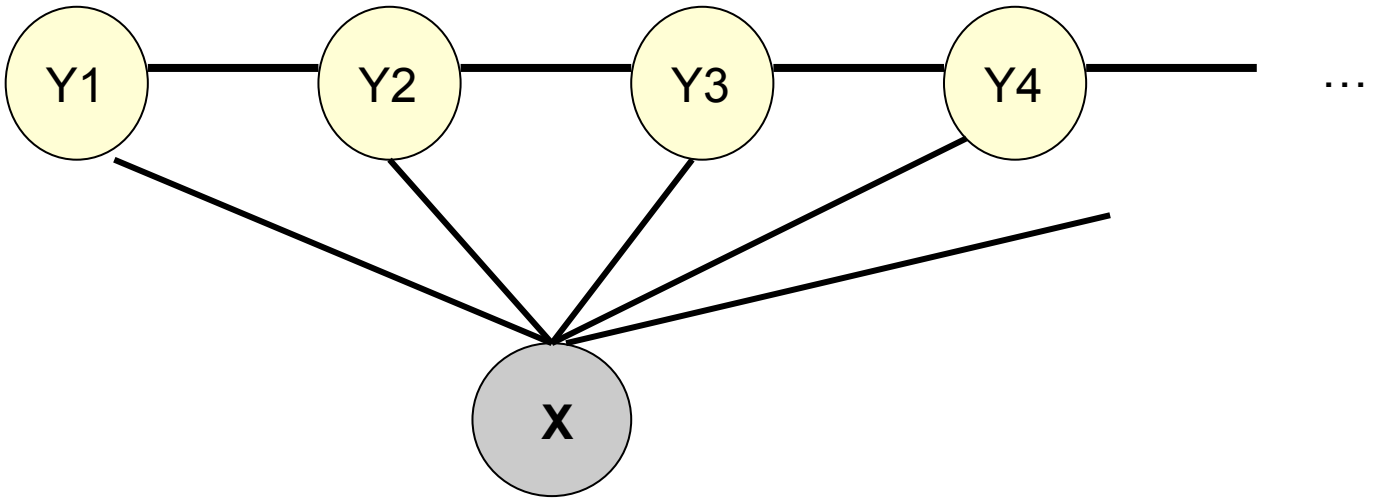

HMM
Y1
Y2
Y3
Y4
…
X1
X2
X3
X4
CRF
Y1
Y2
Y3
Y4
…
X

This is a special case of an *undirected graphical model.*

An undirected graphical model is also called a *Markov network.*

Independencies: every node A is independent of B given the *neighbors of* A (i.e., the nodes directly connected to A):

- For example: <Y3, {Y2,X,Y4},Y1>



Conditional Random Fields: Summary

	Generative; models Pr(X,Y); estimate conditional probs directly	Conditional; models Pr(Y|X); optimize data logliklihood
Instances +labels **x**,y	Naïve Bayes	Logistic regression
Sequences of instances + sequences of labels **x,y**	HMMs	CRFs