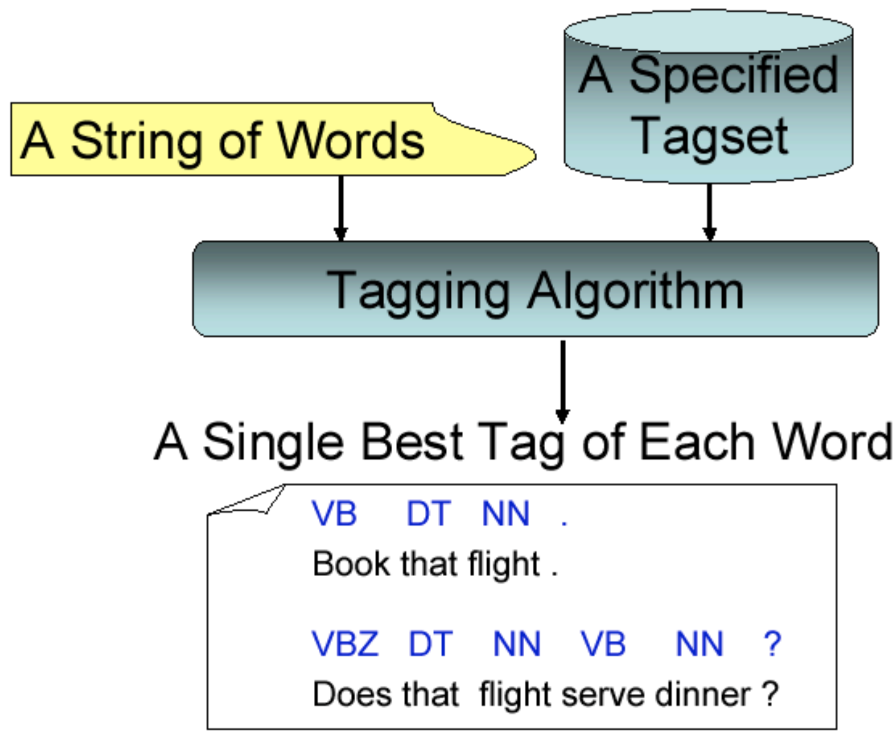
Part-of-Speech Tagging

- Assign grammatical tags to words
- Basic task in the analysis of natural language data
- Phrase identification, entity extraction, etc.
- Ambiguity: “tag” could be a noun or a verb
- “a tag is a part-of-speech label” – context resolves the ambiguity

The Penn Treebank POS Tag Set

Tag	Description	Example	Tag	Description	Example
CC	Coordin. Conjunction	*and, but, or*	SYM	Symbol	+,%, &
CD	Cardinal number	*one, two, three*	TO	"to"	*to*
DT	Determiner	*a, the*	UH	Interjection	*ah, oops*
EX	Existential 'there'	*there*	VB	Verb, base form	*eat*
FW	Foreign word	*mea culpa*	VBD	Verb, past tense	*ate*
IN	Preposition/sub-conj	*of, in, by*	VBG	Verb, gerund	*eating*
JJ	Adjective	*yellow*	VBN	Verb, past participle	*eaten*
JJR	Adj., comparative	*bigger*	VBP	Verb, non-3sg pres	*eat*
JJS	Adj., superlative	*wildest*	VBZ	Verb, 3sg pres	*eats*
LS	List item marker	*1, 2, One*	WDT	Wh-determiner	*which, that*
MD	Modal	*can, should*	WP	Wh-pronoun	*what, who*
NN	Noun, sing. or mass	*llama*	WP$	Possessive wh-	*whose*
NNS	Noun, plural	*llamas*	WRB	Wh-adverb	*how, where*
NNP	Proper noun, singular	*IBM*	$	Dollar sign	$
NNPS	Proper noun, plural	*Carolinas*	#	Pound sign	#
PDT	Predeterminer	*all, both*	“	Left quote	(‘ or “)
POS	Possessive ending	*'s*	”	Right quote	(’ or ”)
PP	Personal pronoun	*I, you, he*	(	Left parenthesis	([, (, {, <)
PP$	Possessive pronoun	*your, one's*	)	Right parenthesis	(],), }, >)
RB	Adverb	*quickly, never*	,	Comma	,
RBR	Adverb, comparative	*faster*	.	Sentence-final punc	(. ! ?)
RBS	Adverb, superlative	*fastest*	:	Mid-sentence punc	(: ; ... – -)
RP	Particle	*up, off*			

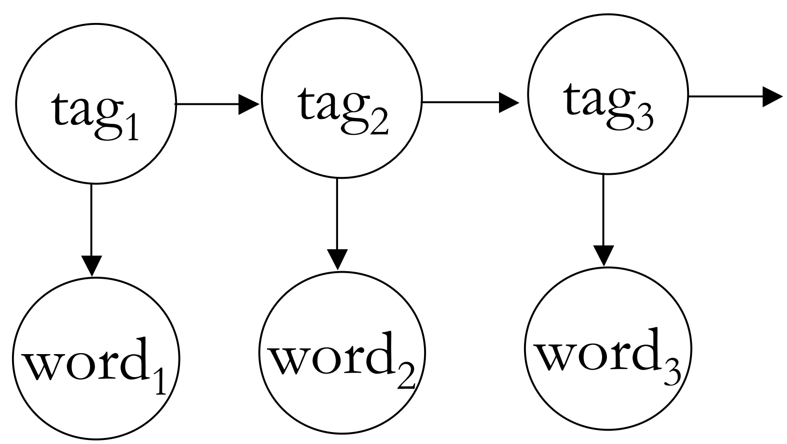
POS Tagging Process



Berlin Chen

POS Tagging Algorithms

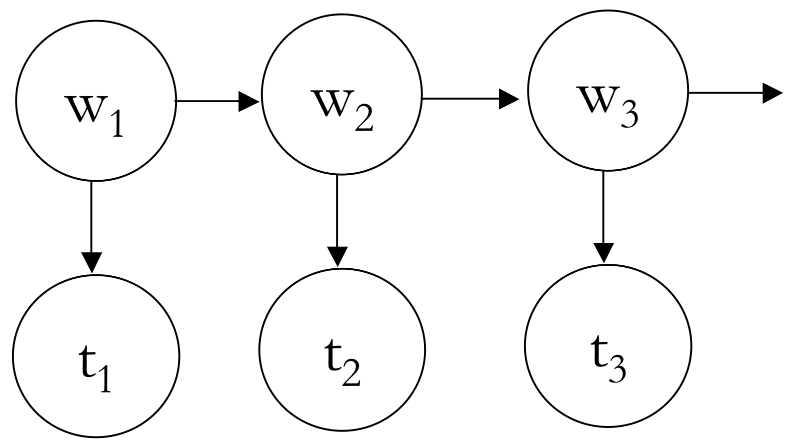
- Rule-based taggers: large numbers of hand-crafted rules
- Probabilistic tagger: used a tagged corpus to train some sort of model, e.g. HMM.



The Brown Corpus

- Comprises about 1 million English words
- HMM's first used for tagging on the Brown Corpus
- 1967. Somewhat dated now.
- British National Corpus has 100 million words

Simple Charniak Model



- What about words that have never been seen before?
- Clever tricks for smoothing the number of parameters (aka priors)

some details…

$$P(t^i \mid w^j) \stackrel{\text{est}}{=} \lambda_1(w^j)\frac{C(t^i, w^j)}{C(w^j)} + \lambda_2(w^j)\frac{\mathrm{C}_n(t^i)}{\mathrm{C}_n()}$$

$C(t^i, w^j)$ number of times word *j* appears with tag *i*

$C(w^j)$ number of times word *j* appears

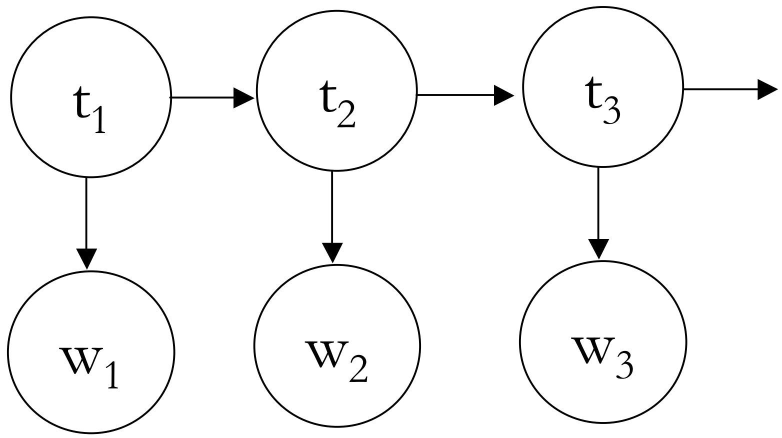
$\mathrm{C}_n(t^i)$ number of times a word that had never been seen with tag *i* gets tag *i*

$\mathrm{C}_n()$ number of such occurrences in total

$$\lambda_1(w^j) = \begin{cases} 1 & \text{if } C(w^j) \geq 1 \\ 0 & \text{otherwise.} \end{cases}$$

Test data accuracy on Brown Corpus = 91.51%

HMM



$$\mathcal{T}(w_{1,n}) = \arg\max_{t_{1,n}} \prod_{i=1}^{n} P(t_i \mid t_{i-1}) P(w_i \mid t_i)$$

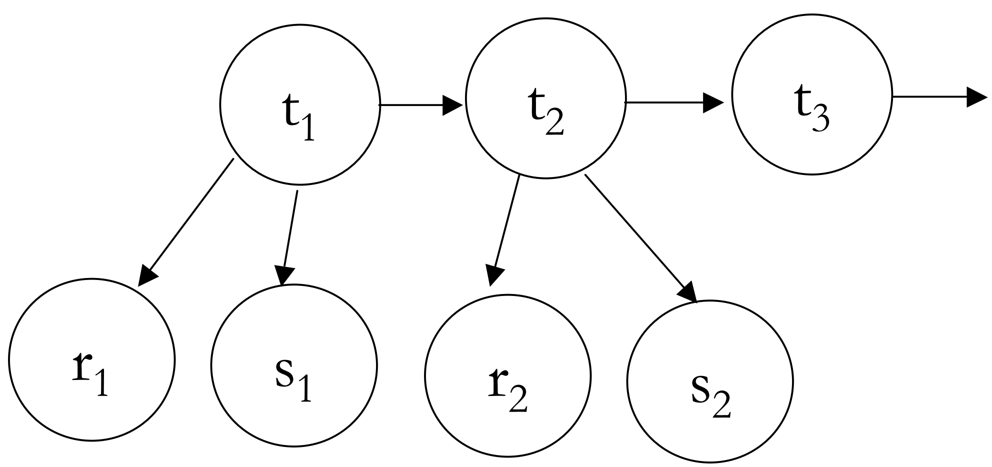
$$= \quad \arg\max_{t_{1,n}} \prod_{i=1}^{n} P(t_i \mid t_{i-1}) \frac{P(t_i \mid w_i)}{P(t_i)}$$

$$P(t_i \mid t_{i-1}) \stackrel{\text{est}}{=} (1-\epsilon)\frac{C(t_{i-1}, t_i)}{C(t_{i-1})} + \epsilon$$

- Brown test set accuracy = 95.97%

Morphological Features

- Knowledge that “quickly” ends in “ly” should help identify the word as an adverb
- “randomizing” -> “ing”
- Split each word into a root (“quick”) and a suffix (“ly”)



Morphological Features

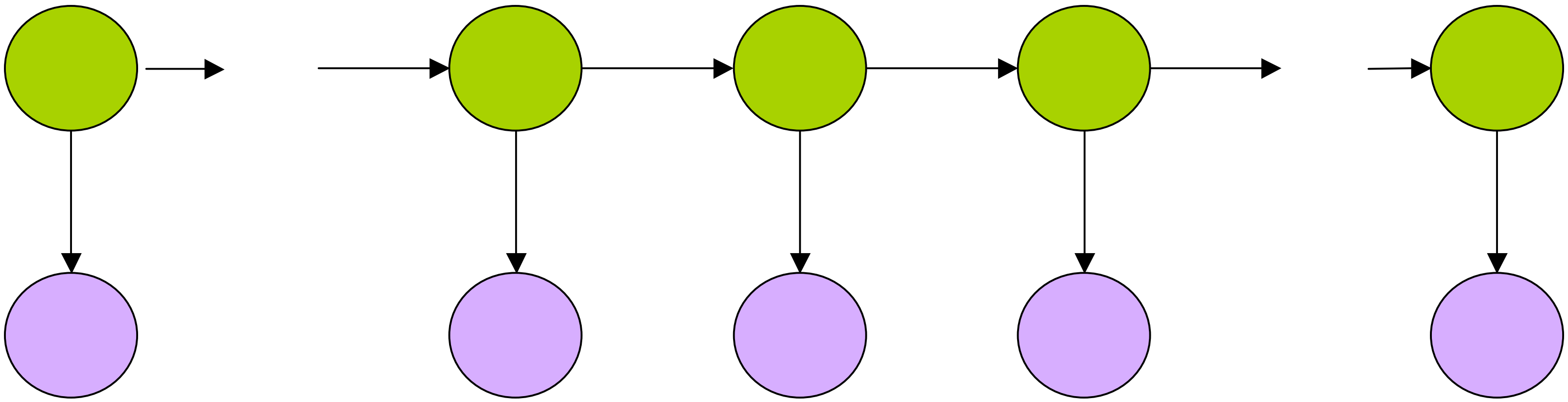
- Typical morphological analyzers produce multiple possible splits
- "Gastroenteritis" ???

$$\mathcal{T}(w_{1,n}) = \arg\max_{t_{1,n}} \sum_{r_{1,n}, s_{1,n}} \prod_{i=1}^{n} P(t_i \mid t_{i-1}) P(s_i \mid t_i) P(r_i \mid t_i) \quad (35)$$

$$= \arg\max_{t_{1,n}} \sum_{r_{1,n}, s_{1,n}} \prod_{i=1}^{n} P(t_i \mid t_{i-1}) P(s_i \mid t_i) \frac{P(r_i) P(t_i \mid r_i)}{P(t_i)} \quad (36)$$

- Achieves 96.45% on the Brown Corpus

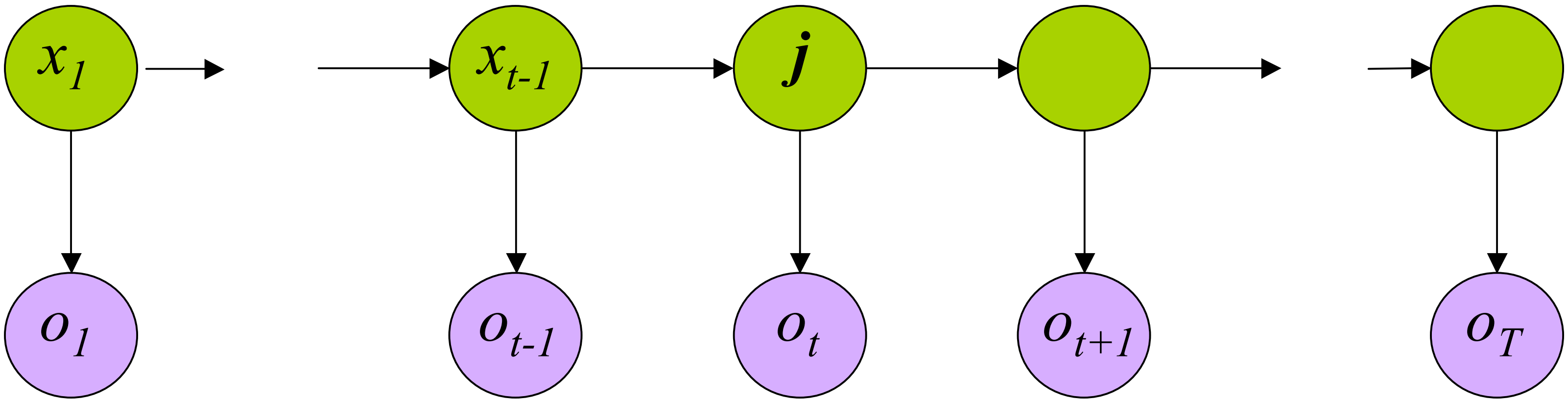
Inference in an HMM



- Compute the probability of a given observation sequence
- $\boxed{\text{Given an observation sequence, compute the most likely hidden state sequence}}$
- Given an observation sequence and set of possible models, which model most closely fits the data?

David Blei

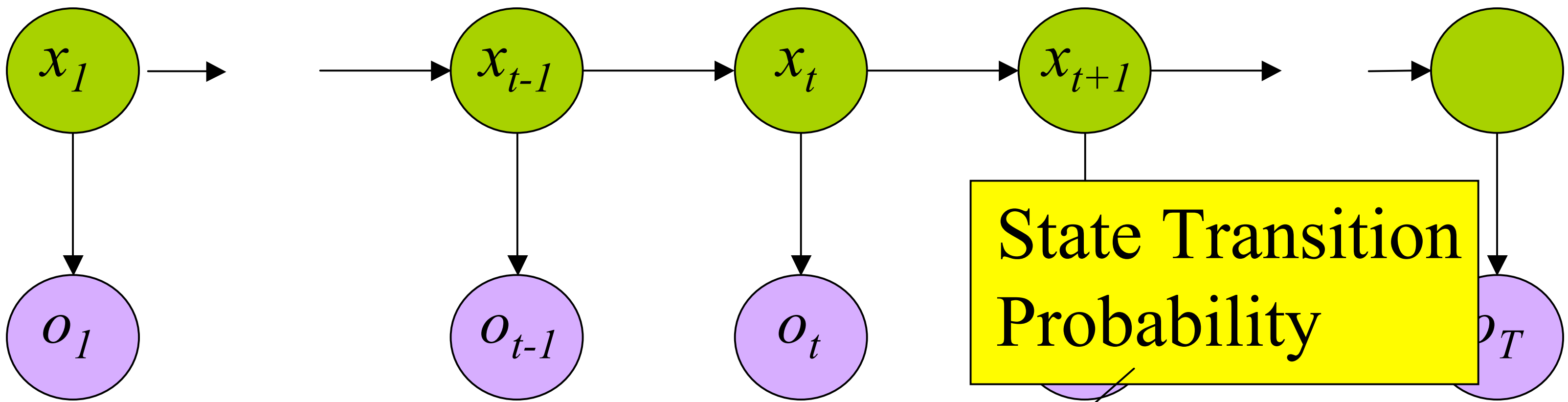
Viterbi Algorithm



$$\delta_j(t) = \max_{x_1 \ldots x_{t-1}} P(x_1 \ldots x_{t-1}, o_1 \ldots o_{t-1}, x_t = j, o_t)$$

The state sequence which maximizes the probability of seeing the observations to time t-1, landing in state j, and seeing the observation at time t

Viterbi Algorithm



State Transition Probability

$$\delta_j(t) = \max_{x_1 \ldots x_{t-1}} P(x_1 \ldots x_{t-1}, o_1 \ldots o_{t-1}, x_t =$$

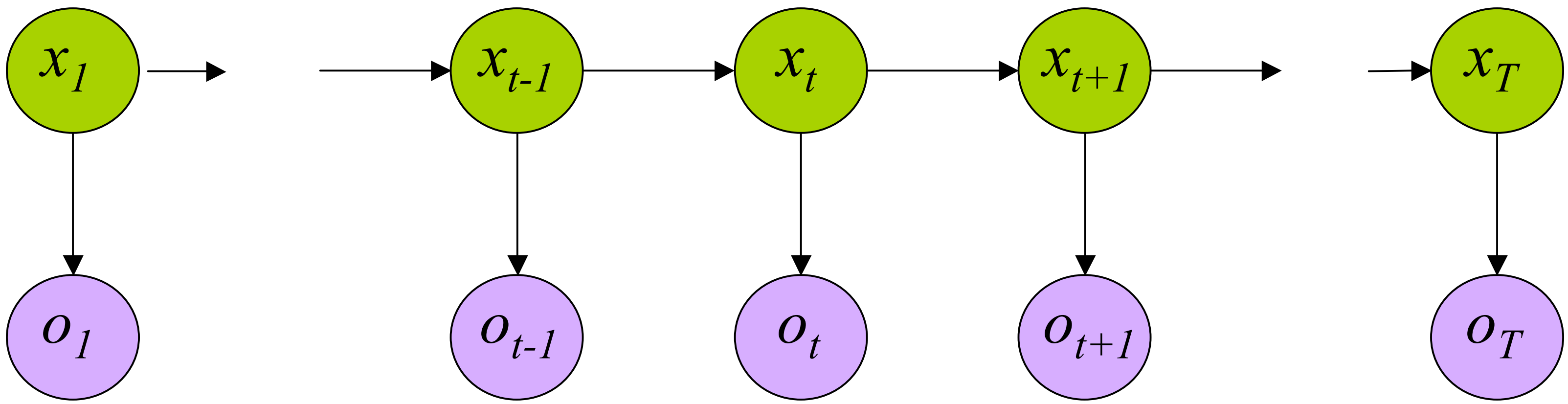
"Emission" Probability

$$\delta_j(t+1) = \max_i \delta_i(t) a_{ij} b_{jo_{t+1}}$$

$$\psi_j(t+1) = \arg\max_i \delta_i(t) a_{ij} b_{jo_{t+1}}$$

Recursive Computation

Viterbi Algorithm



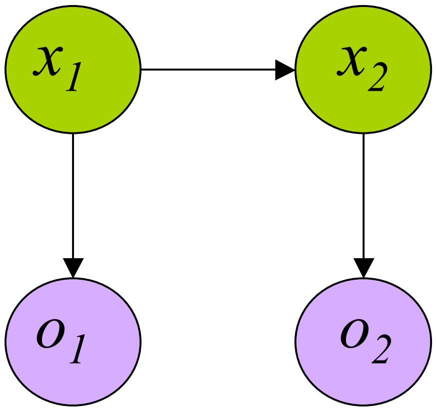
$$\hat{X}_T = \arg\max_j \delta_j(T)$$

$$\hat{X}_t = \psi_{\hat{X}_{t+1}}(t+1)$$

$$P(\hat{X}) = \max_i \delta_i(T)$$

Compute the most likely state sequence by working backwards

Viterbi Small Example



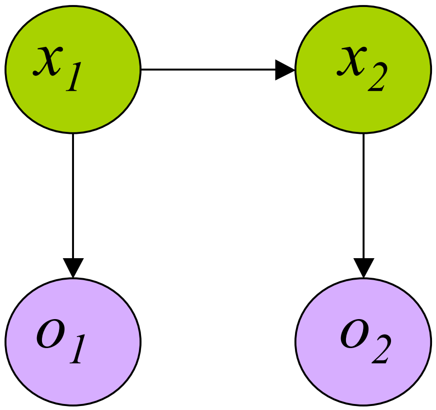
$\Pr(x_1=\mathrm{T}) = 0.2$
$\Pr(x_2=\mathrm{T}|x_1=\mathrm{T}) = 0.7$
$\Pr(x_2=\mathrm{T}|x_1=\mathrm{F}) = 0.1$
$\Pr(o=\mathrm{T}|x=\mathrm{T}) = 0.4$
$\Pr(o=\mathrm{T}|x=\mathrm{F}) = 0.9$
$o_1=\mathrm{T};\ \ o_2=\mathrm{F}$

Brute Force

$\Pr(x_1=\mathrm{T},x_2=\mathrm{T}, o_1=\mathrm{T},o_2=\mathrm{F}) = 0.2 \times 0.4 \times 0.7 \times 0.6 = 0.0336$
$\Pr(x_1=\mathrm{T},x_2=\mathrm{F}, o_1=\mathrm{T},o_2=\mathrm{F}) = 0.2 \times 0.4 \times 0.3 \times 0.1 = 0.0024$
$\Pr(x_1=\mathrm{F},x_2=\mathrm{T}, o_1=\mathrm{T},o_2=\mathrm{F}) = 0.8 \times 0.9 \times 0.1 \times 0.6 = 0.0432$
$\mathbf{\Pr(x_1=F,x_2=F, o_1=T,o_2=F) = 0.8 \times 0.9 \times 0.9 \times 0.1 = 0.0648}$

$$\Pr(X_1,X_2 \mid o_1=\mathrm{T},o_2=\mathrm{F}) \propto \Pr(X_1,X_2 , o_1=\mathrm{T},o_2=\mathrm{F})$$

Viterbi Small Example



$$\hat{X}_2 = \arg\max_j \delta_j(2)$$

$$\delta_j(2) = \max_i \delta_i(1) a_{ij} b_{jo_2}$$

$$\delta_T(1) = \Pr(x_1 = T)\Pr(o_1 = T \mid x_1 = T) = 0.2 \times 0.4 = 0.08$$

$$\delta_F(1) = \Pr(x_1 = F)\Pr(o_1 = T \mid x_1 = F) = 0.8 \times 0.9 = 0.72$$

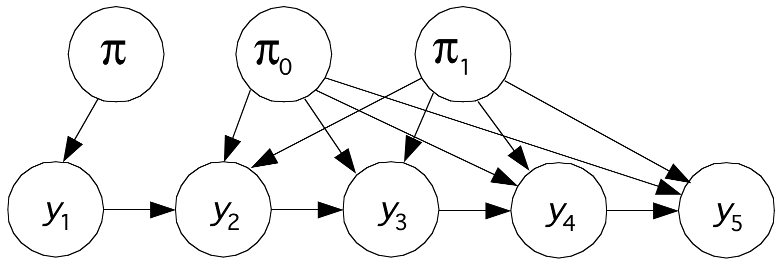
$$\delta_T(2) = \max\left(\delta_F(1) \times \Pr(x_2 = T \mid x_1 = F)\Pr(o_2 = F \mid x_2 = T), \delta_T(1) \times \Pr(x_2 = T \mid x_1 = T)\Pr(o_2 = F \mid x_2 = T)\right)$$

$$= \max(\underline{0.72 \times 0.1 \times 0.6}, 0.08 \times 0.7 \times 0.6) = 0.0432$$

$$\delta_F(2) = \max\left(\delta_F(1) \times \Pr(x_2 = F \mid x_1 = F)\Pr(o_2 = F \mid x_2 = F), \delta_T(1) \times \Pr(x_2 = F \mid x_1 = T)\Pr(o_2 = F \mid x_2 = F)\right)$$

$$= \max(\underline{0.72 \times 0.9 \times 0.1}, 0.0.08 \times 0.3 \times 0.1) = 0.0648$$

Bayesian Analysis of a Markov Chain



$$y_i \text{ take values in } \{0,1\}, i = 1, \ldots, 5$$

$$\pi = p(y_1 = 1)$$

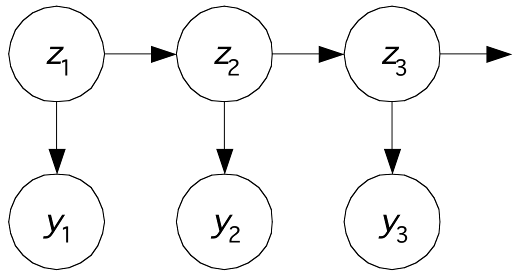
$$\pi_0 = p(y_i = 1 | y_{i-1} = 0) \quad \pi_1 = p(y_i = 1 | y_{i-1} = 1)$$

$$\pi, \pi_0, \pi_1 \overset{\text{iid}}{\sim} \text{Beta}(1,1)$$

Suppose we observe $\{1,1,1,1,1\}$.

Then $\pi \sim \text{Beta}(2,1)$, $\pi_0 \sim \text{Beta}(1,1)$ and $\pi_1 \sim \text{Beta}(5,1)$

Bayesian Analysis of a HMM



- Widely used in speech recognition, finance, bioinformatics, etc.
- The *y*'s are observed but the *z*'s (discrete) are not
- Combines a first-order dependence structure with a mixture model.

Bayesian HMM (continued)

Suppose z_i take values in $\{0, 1\}, i = 1, \ldots, n$

$$y_i \sim N(\mu_{z_i}, 1)$$

$$\begin{aligned} [\theta|y] \quad &\propto \quad [\mu_0][\mu_1][\pi_0][\pi_1][y_1|1] \\ &\times \left(\sum_{i_2=0}^{1} \sum_{i_3=0}^{1} \cdots \sum_{i_n=0}^{1} \pi_{1\to i_2}[y_2|i_2]\pi_{i_2\to i_3}[y_3|i_3] \ldots \pi_{i_{n-1}\to i_n}[y_n|i_n] \right) \end{aligned}$$

this is generally intractable but the conditionals are OK:

$[\theta|y, z] \propto$ <u>depends on the priors…</u>

$$\begin{aligned} [z_i|y, \theta, z_{-i}] \quad &\propto \quad [z_i|z_{i-1}, z_{i+1}, y_i, \theta] \\ &\propto \quad [y_i|z_i, \theta][z_{i+1}|z_i, \theta][z_i|z_{i-1}, \theta] \end{aligned}$$

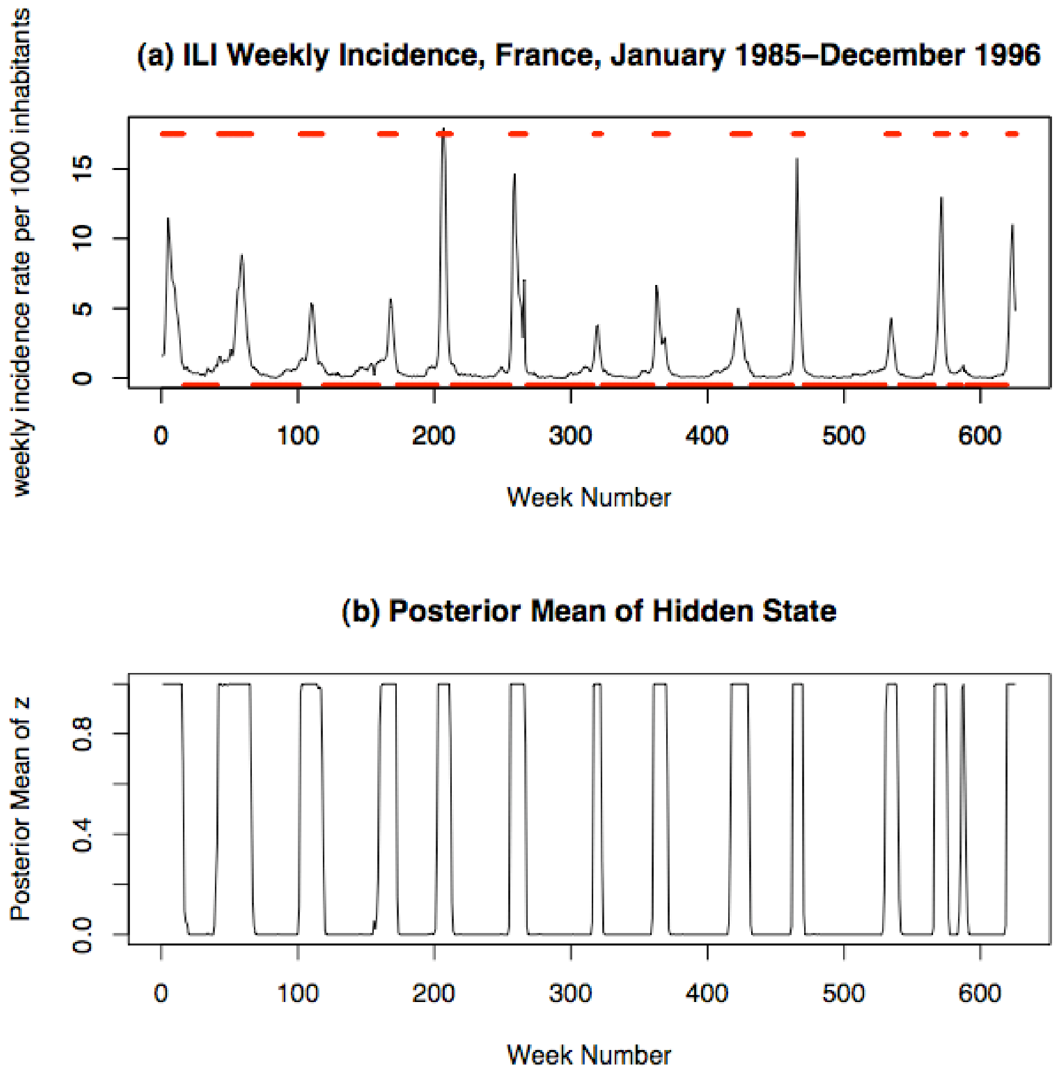


Figure 3: *The French ILI data. The upper figure (a) shows the incidence rates per 1,000 inhabitants. The lower figure (b) shows the posterior mean of the hidden state from a Gaussian two-state HMM. The horizontal line segments in the upper figure correspond to time periods where the posterior mean of the hidden state exceeds 0.5.*

Serfling's method

$$\mu_j(t) = \gamma_j + \beta_j t + \delta_j \cos\left(\frac{2\pi t}{r}\right) + \epsilon_j \left(\frac{2\pi t}{r}\right)$$

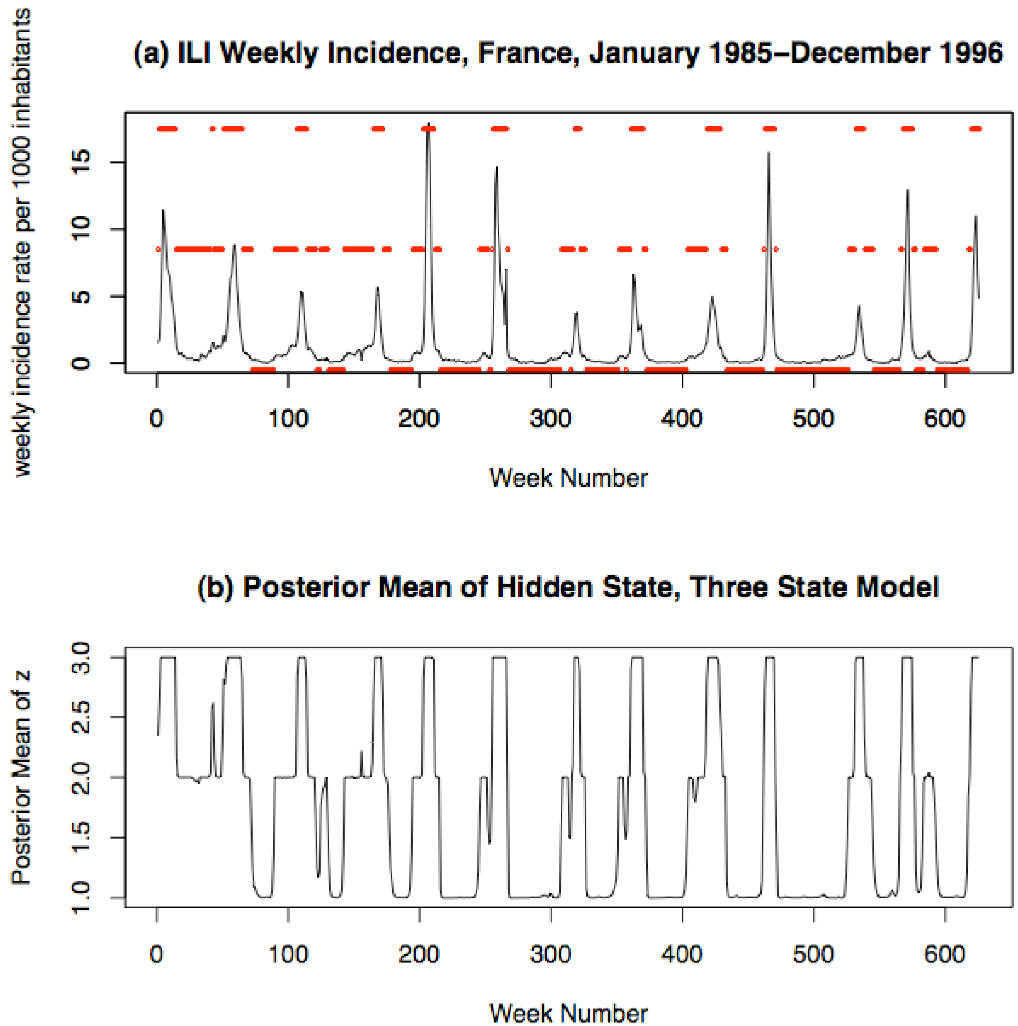


Figure 7: *The French ILI data. The upper figure (a) shows the incidence rates per 1,000 inhabitants. The lower figure (b) shows the posterior mean of the hidden state from a Gaussian three-state HMM. The horizontal line segments in the upper figure correspond to the three different states.*

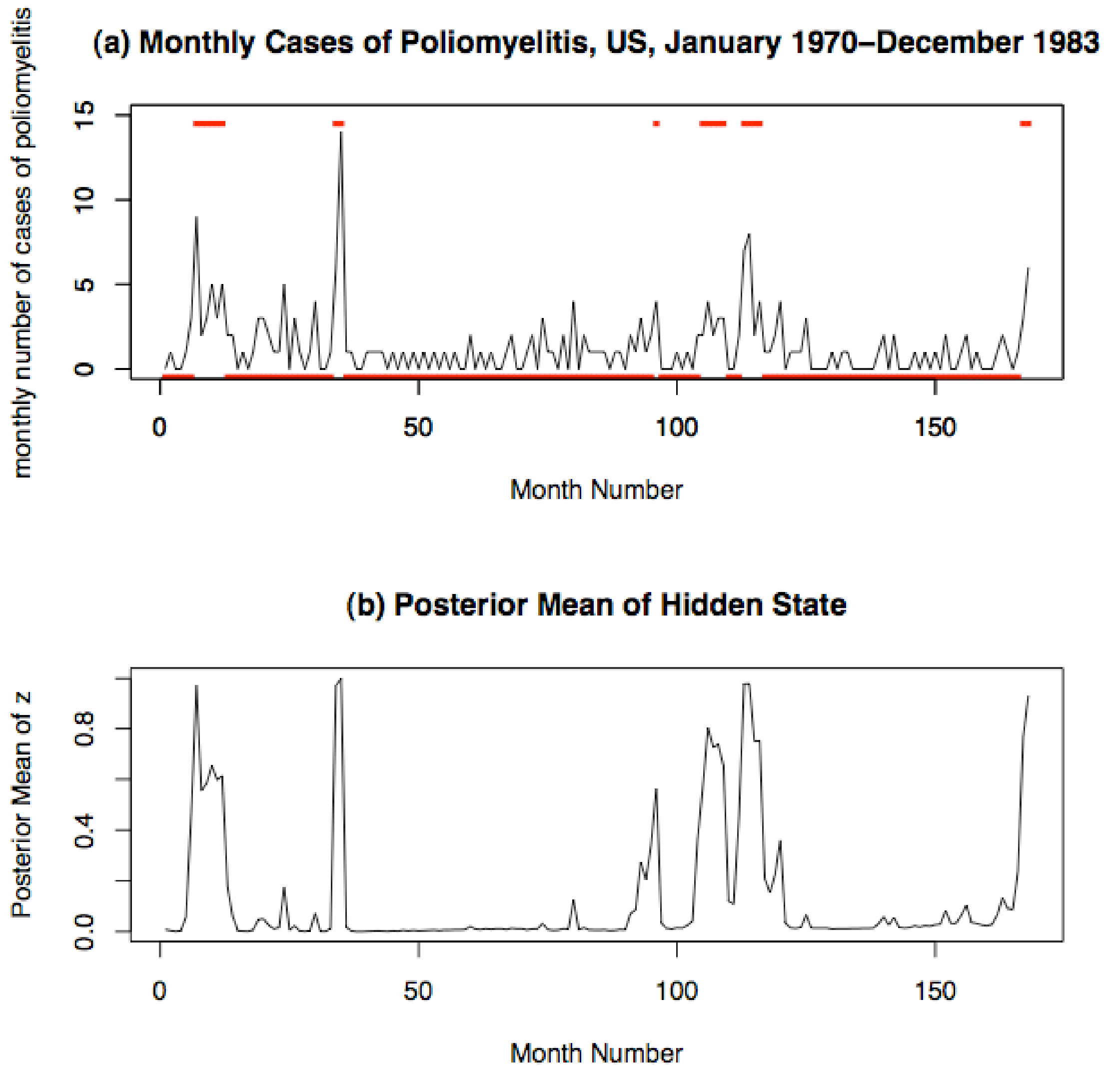
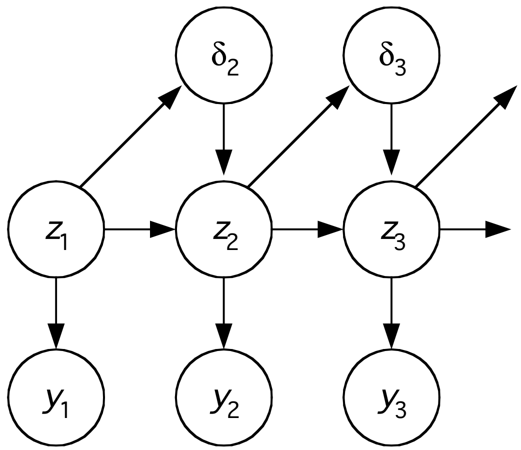


Figure 6: *The US Poliomyelitis data. The upper figure (a) shows the total reported monthly cases. The lower figure (b) shows the posterior mean of the hidden state from a Poisson two-state HMM. The horizontal line segments in the upper figure correspond to time periods where the posterior mean of the hidden state exceeds 0.5.*

Bayesian HMM for High-Frequency Data

- Observations may not arrive regularly
- Elapsed time between observations may be related to state
- Finance: tick-level stock data
- Molecular biology: single molecule experiments
- Assume now that z's follow a continuous-time first-order Markov chain

HF-HMM



$$[y_t | z_t = i] \sim \mathcal{N}(0, \sigma_i^2), i = 0, \ldots, K-1.$$

$$[\delta_t | z_{t-1} = i] \sim geometric(p_i), i = 0, \ldots, K-1$$

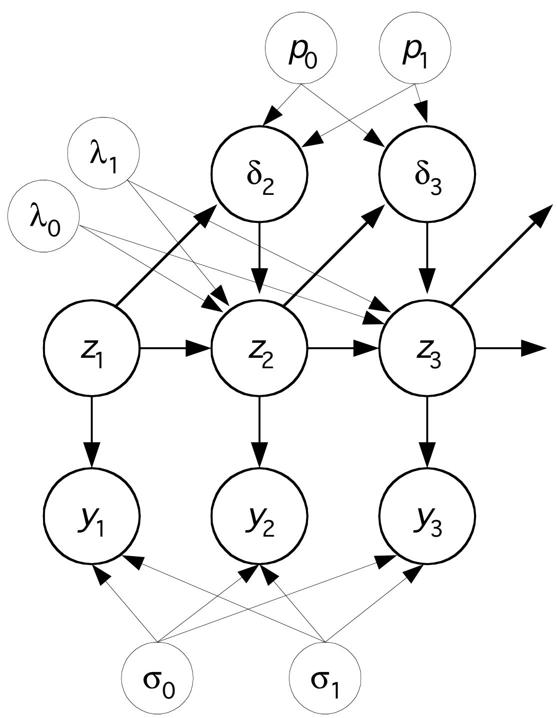
For K=2, suppose the time z stays in state i is exp(λ_i). Then:

$$p(z_t = 0 | z_{t-1} = 0, \delta_t = t) = \frac{\lambda_1}{\lambda_0 + \lambda_1} + \frac{\lambda_0}{\lambda_0 + \lambda_1} \exp^{-(\lambda_0 + \lambda_1)t}$$

For K>2, from state i, z transitions to state i+1 with probability βi and state i-1 with probability 1- βi (i.e. birth & death, reflecting boundaries)

eigendecomposition…

HF-HMM Priors



$$
\begin{aligned}
[\sigma_i^2] &\sim \mathrm{Inv}\!-\!\chi^2(\nu_0, \sigma_0^2), i = 0, \ldots, K-1 \\
[p_i] &\sim \mathrm{Beta}(\alpha_p, \beta_p), i = 0, \ldots, K-1 \\
[\lambda_i] &\sim \Gamma(\alpha_\lambda, \beta_\lambda), i = 0, \ldots, K-1 \\
[\beta_i] &\sim \mathrm{Beta}(\alpha_\beta, \beta_\beta), i = 1, \ldots, K-2
\end{aligned}
$$

Posteriors not available in closed-form…

HF-HMM Gibbs Sampler

$$
\begin{aligned}
[z_t|-] &\propto [z_t|z_{t-1}, \delta_t, \lambda_0, \lambda_1]\,[z_{t+1}|z_t, \delta_{t+1}, \lambda_0, \lambda_1]\,[\delta_{t+1}|z_t, p_0, p_1]\,[y_t|z_t, \sigma_0, \sigma_1] \\
\left[\sigma_i^2|-\right] &\propto [\sigma_i] \prod_{\substack{t=1 \\ z_t=i}}^{n} [y_t|z_t, \sigma_i] \\
[p_i|-] &\propto [p_i] \sum_{\substack{t=2 \\ z_{t-1}=i}}^{n} [\delta_t|z_{t-1}, p_i] \\
[\boldsymbol{\lambda}, \boldsymbol{\beta}|-] &\propto [\boldsymbol{\lambda}]\,[\boldsymbol{\beta}] \prod_{t=2}^{n} [z_t|z_{t-1}, \delta_t, \boldsymbol{\lambda}, \boldsymbol{\beta}]
\end{aligned}
$$

Use a Metropolis step for this one

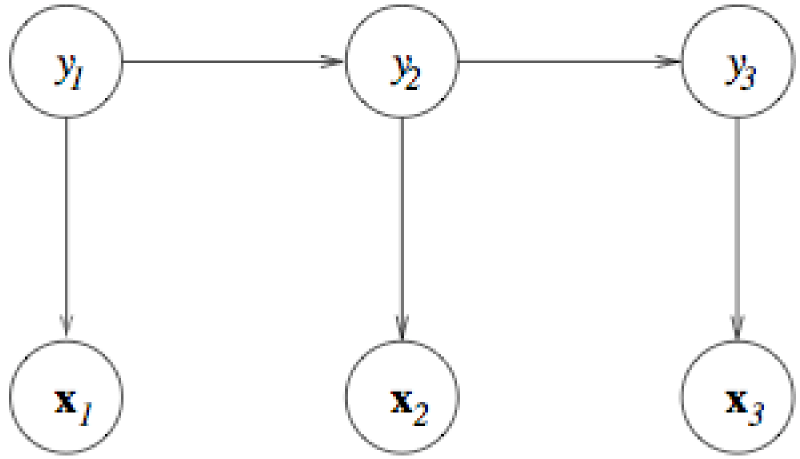
Metropolis Within Gibbs

Let $\mathbf{\Lambda} \equiv (\boldsymbol{\lambda}, \boldsymbol{\beta})$

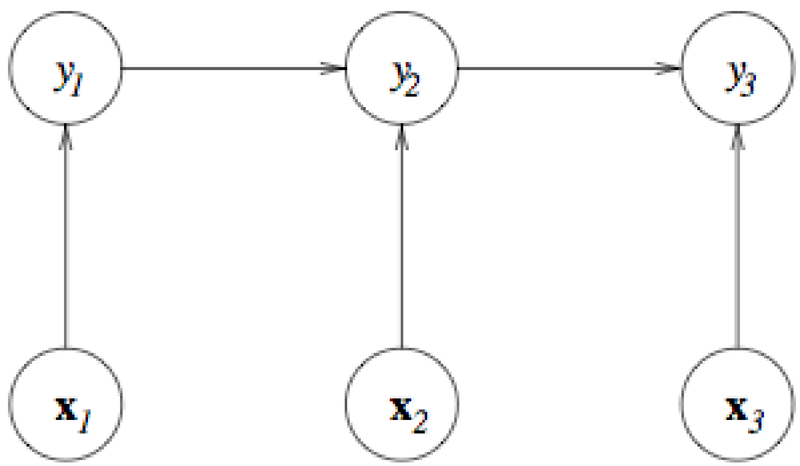
Generate a candidate from: $J(\mathbf{\Lambda}, \mathbf{\Lambda}')$

Accept with probability:

$$\alpha(\mathbf{\Lambda}, \mathbf{\Lambda}' | \mathbf{z}, \boldsymbol{\delta}) = \min\{1, \frac{[\mathbf{\Lambda}'] \prod_{t=2}^{n} [z_t | z_{t-1}, \delta_t, \mathbf{\Lambda}']}{[\mathbf{\Lambda}] \prod_{t=2}^{n} [z_t | z_{t-1}, \delta_t, \mathbf{\Lambda}]} \times \frac{J(\mathbf{\Lambda}', \mathbf{\Lambda})}{J(\mathbf{\Lambda}, \mathbf{\Lambda}')}\}.$$



<u>HMM</u>



<u>MEMM</u>

maximum entropy markov model

MEMM

- MEMM learns a single multiclass Logistic regression model for $y_i \mid y_{i-1}, x_i$

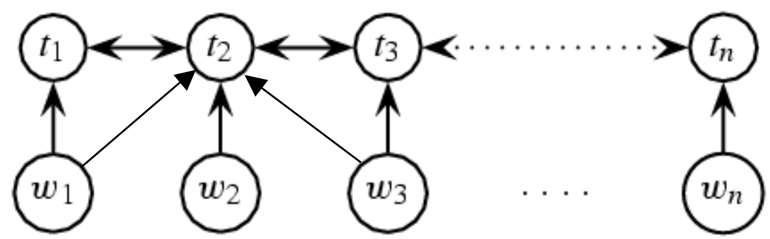
- Predict y1 from x1, then y2 from y1 and x2, etc.

- No reason for the features not to include x_{i-1}, x_{i+1}, etc.

t_i	t_{i-1}	f_1	f_2	f_3	…	f_d
PER	LOC	T	1	2.7		0
…						
…						

Dependency Network

- Toutanova et al., 2003, use a "dependency network" and richer feature set

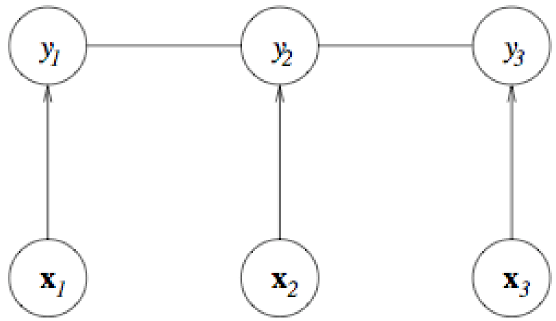


•Idea: using the "next" tag as well as the "previous" tag should improve tagging performance

•Need modified Viterbi to find most likely sequence

t_i	t_{i-1}	t_{i+1}	f_1	f_2	…	f_d
PER	LOC	PER	1	2.7		0
…						
…						

Conditional Random Fields



- Dependency network does consider the tag sequence in its entirety
- CRF's optimize model parameters with respect to the entire sequence
- More expensive optimization; increased flexibility and accuracy

From Logistic Regression to CRF

scalar

- Logistic regression:

$$p(y|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{ \lambda_y + \sum_{j=1}^{K} \lambda_{y,j} x_j \right\}$$

- Or

$$p(y|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y, \mathbf{x}) \right\}$$

- Linear chain CRF:

$$p(y \mid x) = \frac{1}{Z(x)} \prod_t \exp\left\{ \sum_{k=1}^{K} \lambda_k f_k(y_t, y_{t-1}, x) \right\}$$

vector

CRF Parameter Estimation

- Conditional log likelihood:

$$\ell(\theta) = \sum_{i=1}^{N} \log p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)})$$

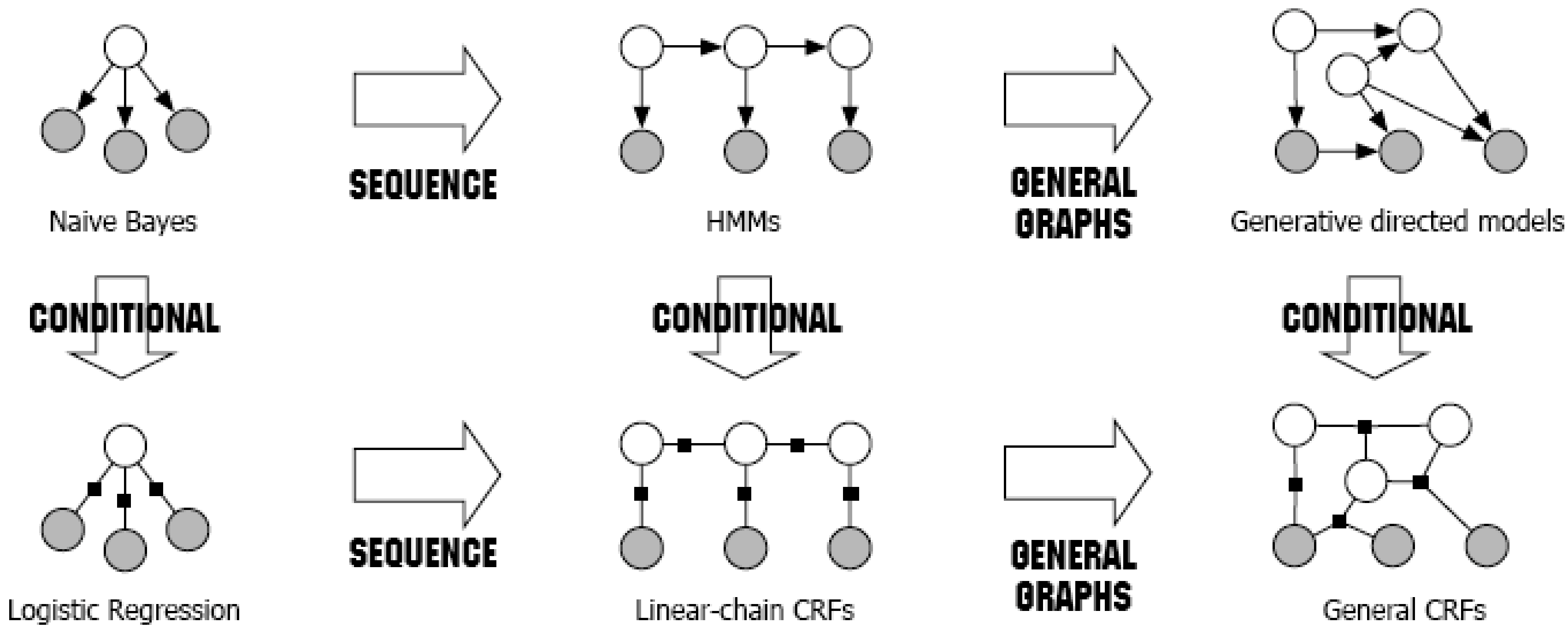
$$= \sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}),$$

- Regularized log likelihood:

$$\sum_{i=1}^{N}\sum_{t=1}^{T}\sum_{k=1}^{K} \lambda_k f_k(y_t^{(i)}, y_{t-1}^{(i)}, \mathbf{x}_t^{(i)}) - \sum_{i=1}^{N} \log Z(\mathbf{x}^{(i)}) - \sum_{k=1}^{K} \frac{\lambda_k^2}{2\sigma^2}$$

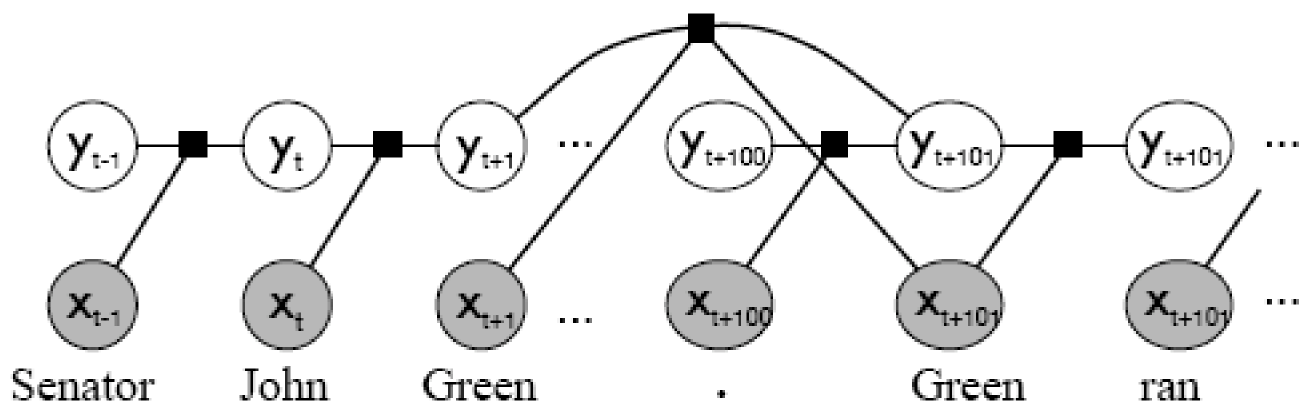
- Conjugate gradient, BFGS, etc.

- POS Tagging, 45 tags, 10^6 words = 1 week



Sutton and McCallum (2006)

Skip-Chain CRF's



Bock's Results: POS Tagging

Feature Description	**Node Feature**	**Edge Feature**
Current word	X	X
Previous word	X	X
Previous word and current word	X	X
Next word	X	X
Current word and next word	X	
Current word shape	X	
Previous word shape	X	
Previous word shape and current word shape	X	
Next word shape	X	
Current word shape and next word shape	X	
Previous word shape, current word shape, and next word shape	X	
Word contains a digit	X	
Two letter word prefix	X	
Three letter word prefix	X	
Four letter word prefix	X	
Five letter word prefix	X	
Two letter word suffix	X	
Three letter word suffix	X	
Four letter word suffix	X	
Five letter word suffix	X	
Word contains a capital letter	X	
Word is followed within three words by Co/Inc/Ltd/Corp	X	
Word contains a hyphen	X	

Model	**Tag Accuracy**	**LBFGS Iterations**	**Training Time (h:mm)**
ME Classifier	96.71%	128	2:09
MEMM	96.81%	194	2:56
CRF	97.00%	207	5:56

Penn Treebank

Bock's Results: Named Entity

Feature Description	Node Feature	Edge Feature
Current word	X	X
Previous word	X	X
Previous word and current word	X	X
Next word	X	X
Current word and next word	X	
Current word shape	X	
Previous word shape	X	
Previous word shape and current word shape	X	
Next word shape	X	
Current word shape and next word shape	X	
Previous word shape, current word shape, and next word shape	X	
Word contains a digit	X	
Two letter word prefix	X	
Three letter word prefix	X	
Four letter word prefix	X	
Five letter word prefix	X	
Two letter word suffix	X	
Three letter word suffix	X	
Four letter word suffix	X	
Five letter word suffix	X	
Word contains a capital letter	X	
Word is followed within three words by Co/Inc/Ltd/Corp	X	
Word contains a hyphen	X	

Model	F1	LBFGS Iterations	Training Time (h:mm)
ME Classifier	84.68%	114	0:08
MEMM	88.03%	106	0:07
CRF	89.38%	166	0:26

CoNLL-03 task