CS388: Natural Language Processing

Lecture 15: HMMs, POS

Greg Durrett

TEXAS
The University of Texas at Austin

Administrivia

- Project 3 due Thursday

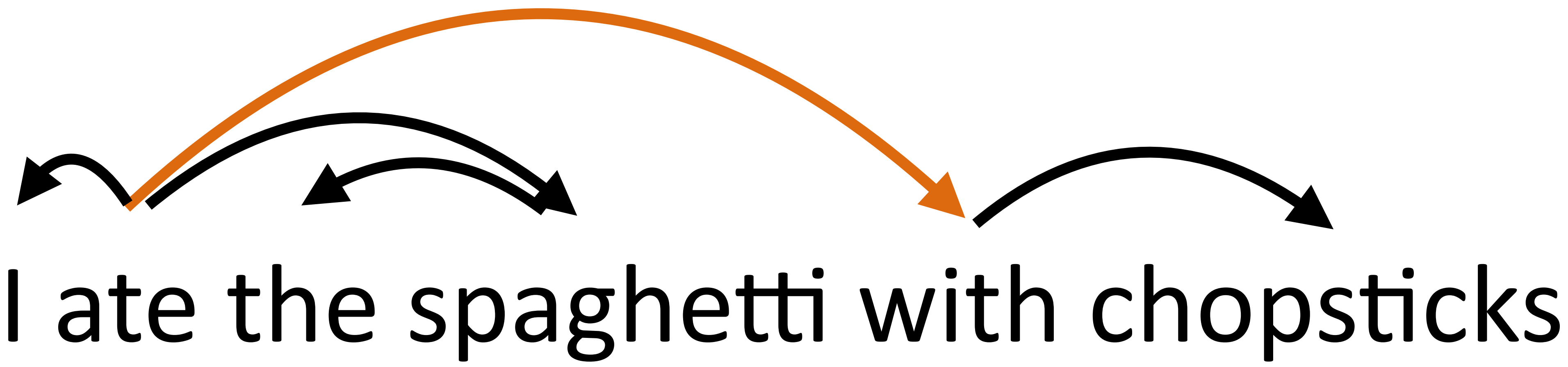
This Lecture

- Part-of-speech tagging
- Hidden Markov Models, parameter estimation
- Viterbi algorithm
- POS taggers
- NER, CRFs, state-of-the-art in sequence modeling

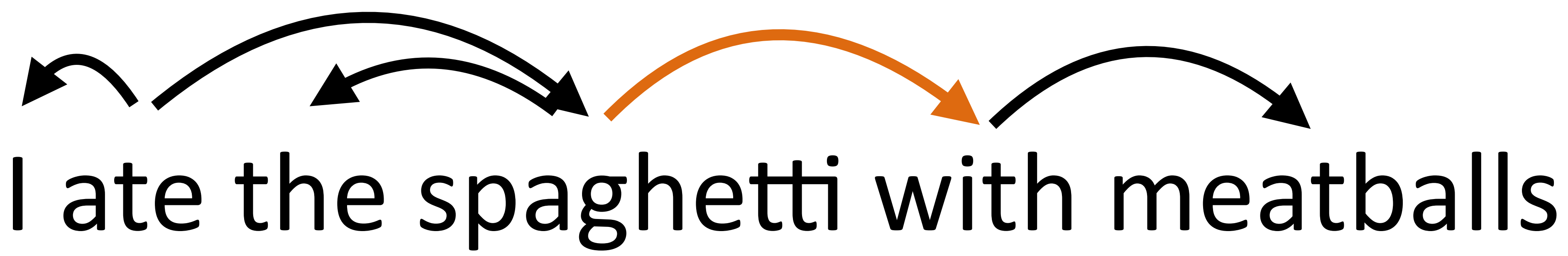
Where are we in the course?

- Next three lectures: structured prediction. Produce representations of language as sequences and trees

- Language has hierarchical structure:



I ate the spaghetti with chopsticks



I ate the spaghetti with meatballs

- Understanding syntax fundamentally requires trees — the sentences have the same shallow analysis. But the first step we'll take towards understanding this is understanding **parts of speech**

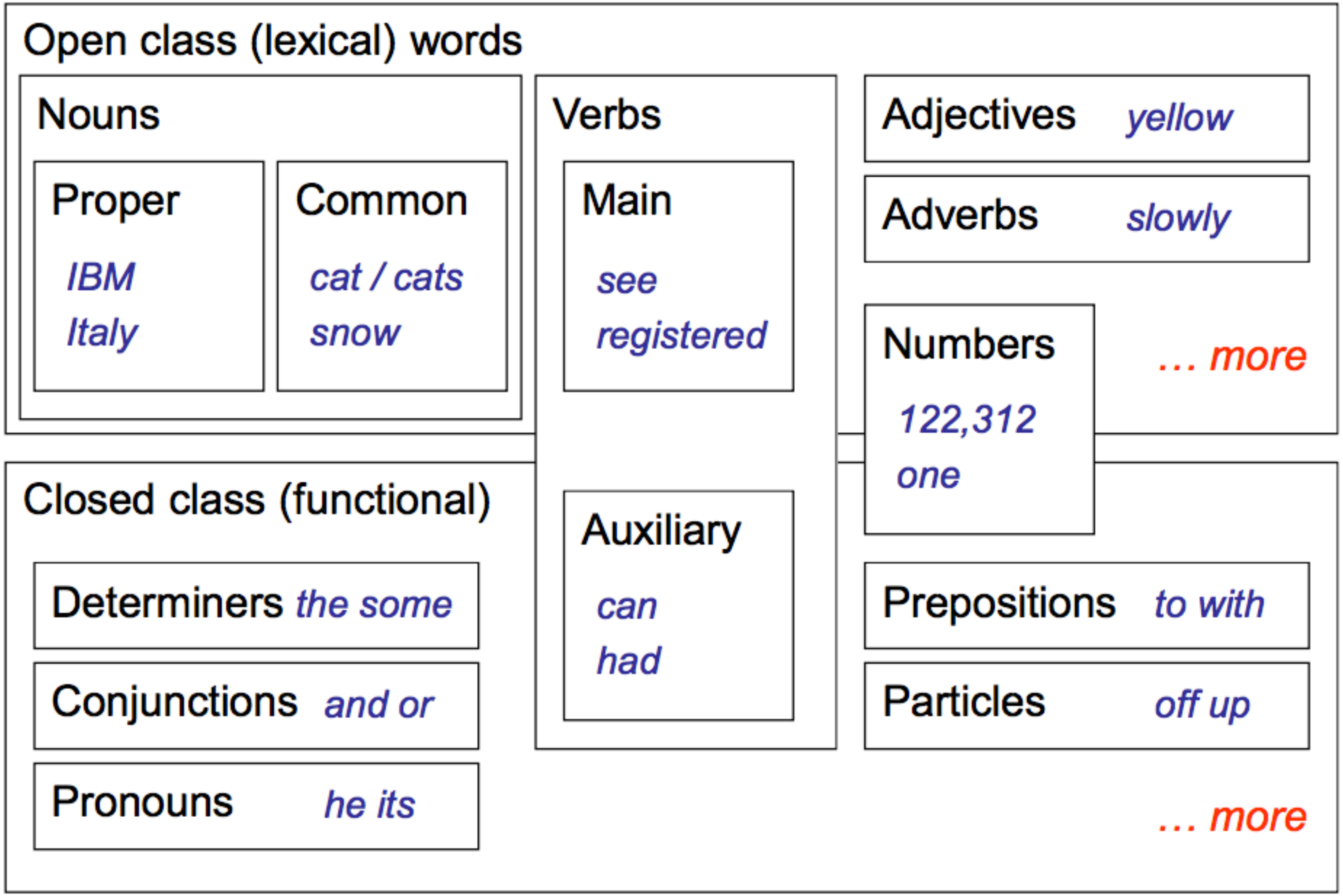
NN NNS VBZ NNS
Teacher strikes idle kids

VBP
I **record** the video

NN
I listen to the **record**

POS Tagging

POS Tagging



Open class (lexical) words

Nouns

Proper *IBM* *Italy*

Common *cat / cats* *snow*

Verbs

Main *see* *registered*

Adjectives *yellow*

Adverbs *slowly*

Numbers *122,312* *one*

… more

Closed class (functional)

Determiners *the some*

Conjunctions *and or*

Pronouns *he its*

Auxiliary *can* *had*

Prepositions *to with*

Particles *off up*

… more

Slide credit: Dan Klein

POS Tagging

VBD		VB			
VBN	VBZ	VBP	VBZ		
NNP	NNS	NN	NNS	CD	NN
Fed	raises	interest	rates	0.5	percent

VBD		VB			
VBN	VBZ	VBP	VBZ		
NNP	NNS	NN	NNS	CD	NN
Fed	raises	interest	rates	0.5	percent

I hereby increase interest rates 0.5%

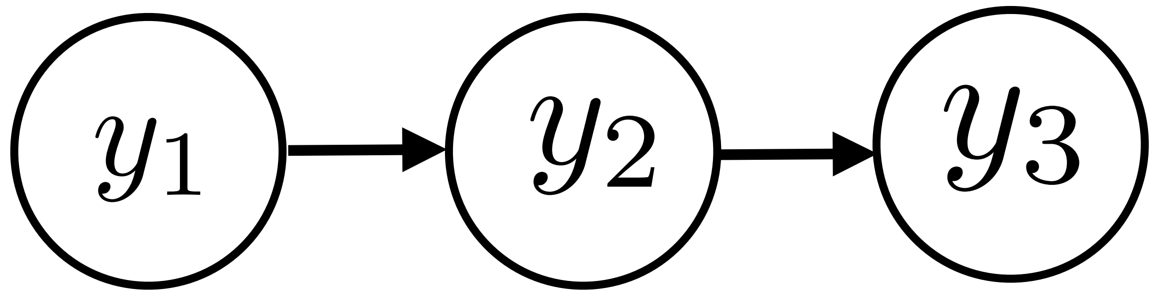
I'm 0.5% interested in the Fed's raises!

- Other paths are also plausible but even more semantically weird...
- What governs the correct choice? Word + context
  - Word identity: most words have <=2 tags, many have one (*percent*, *the*)
  - Context: nouns start sentences, nouns follow verbs, etc.

Hidden Markov Models

Hidden Markov Models

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$
- Model the sequence of tags ***y*** over words ***x*** as a Markov process
- Markov property: future is conditionally independent of the past given the present



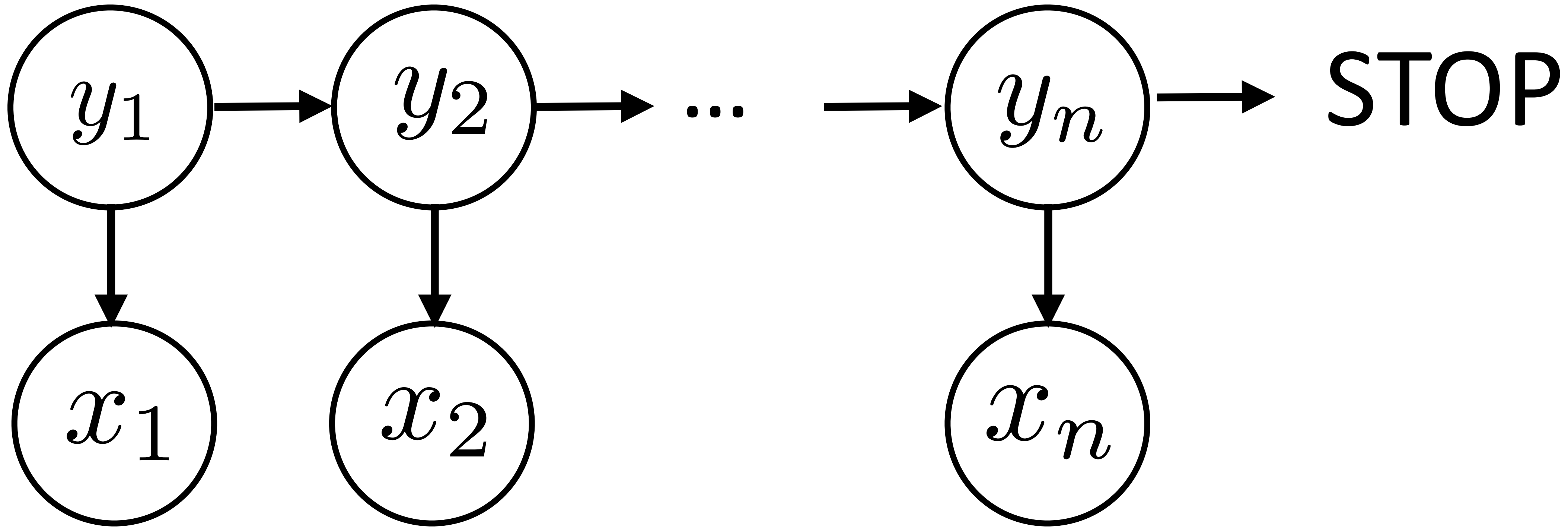
$$P(y_3|y_1, y_2) = P(y_3|y_2)$$

- If ***y*** are tags, this roughly corresponds to assuming that the next tag only depends on the current tag, not anything before

Hidden Markov Models

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$

y ∈ T = set of possible tags (including STOP);
x ∈ V = vocab of words

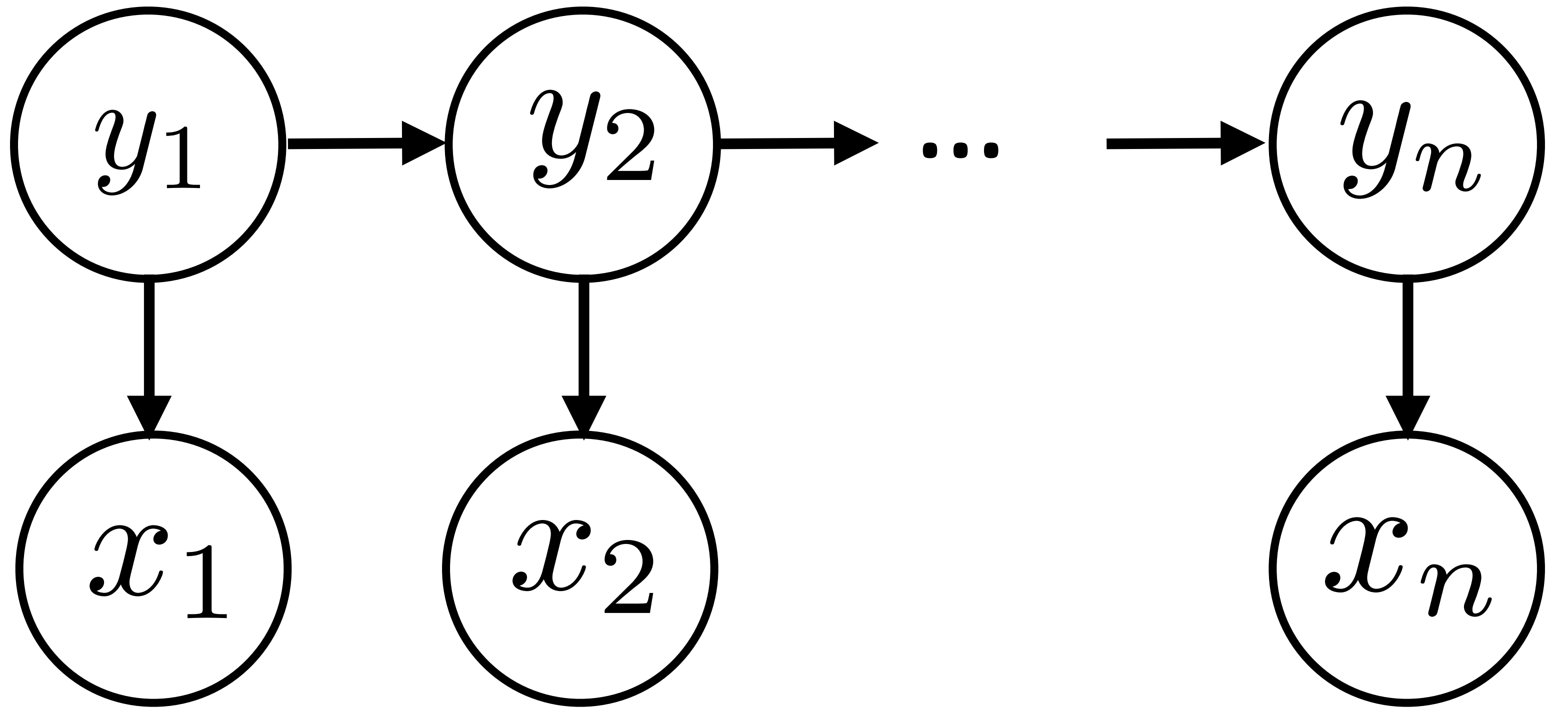


$$P(\mathbf{y}, \mathbf{x}) = \underbrace{P(y_1)}_{\text{Initial distribution}} \underbrace{\prod_{i=2}^{n} P(y_i|y_{i-1})}_{\text{Transition probabilities}} \underbrace{\prod_{i=1}^{n} P(x_i|y_i)}_{\text{Emission probabilities}}$$

- Observation (*x*) depends only on current state (*y*)

HMMs: Parameters

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$



$$P(\mathbf{y}, \mathbf{x}) = P(y_1) \prod_{i=2}^{n} P(y_i | y_{i-1}) \prod_{i=1}^{n} P(x_i | y_i)$$

- Initial distribution: |T| x 1 vector (distribution over initial states)
- Emission distribution: |T| x |V| matrix (distribution over words per tag)
- Transition distribution: |T| x |T| matrix (distribution over next tags per tag)

HMMs Example

Tags = {N , V, STOP}

Vocabulary = {they, can, fish}

Initial

y_1	N	1.0
	V	0
	STOP	0

Transition

		y_i		
		N	V	STOP
y_{i-1}	N	1/5	3/5	1/5
	V	1/5	1/5	3/5

Emission

		x_i		
		they	can	fish
y_i	N	1	0	0
	V	0	1/2	1/2

Transitions in POS Tagging

VBD		VB			
VBN	VBZ	VBP	VBZ		
NNP	NNS	NN	NNS	CD	NN
Fed	raises	interest	rates	0.5	percent

- $P(y_1 = \mathrm{NNP})$ likely because start of sentence
- $P(y_2 = \mathrm{VBZ} | y_1 = \mathrm{NNP})$ likely because verb often follows noun
- $P(y_3 = \mathrm{NN} | y_2 = \mathrm{VBZ})$: direct object can follow verb

- How are these probabilities learned?

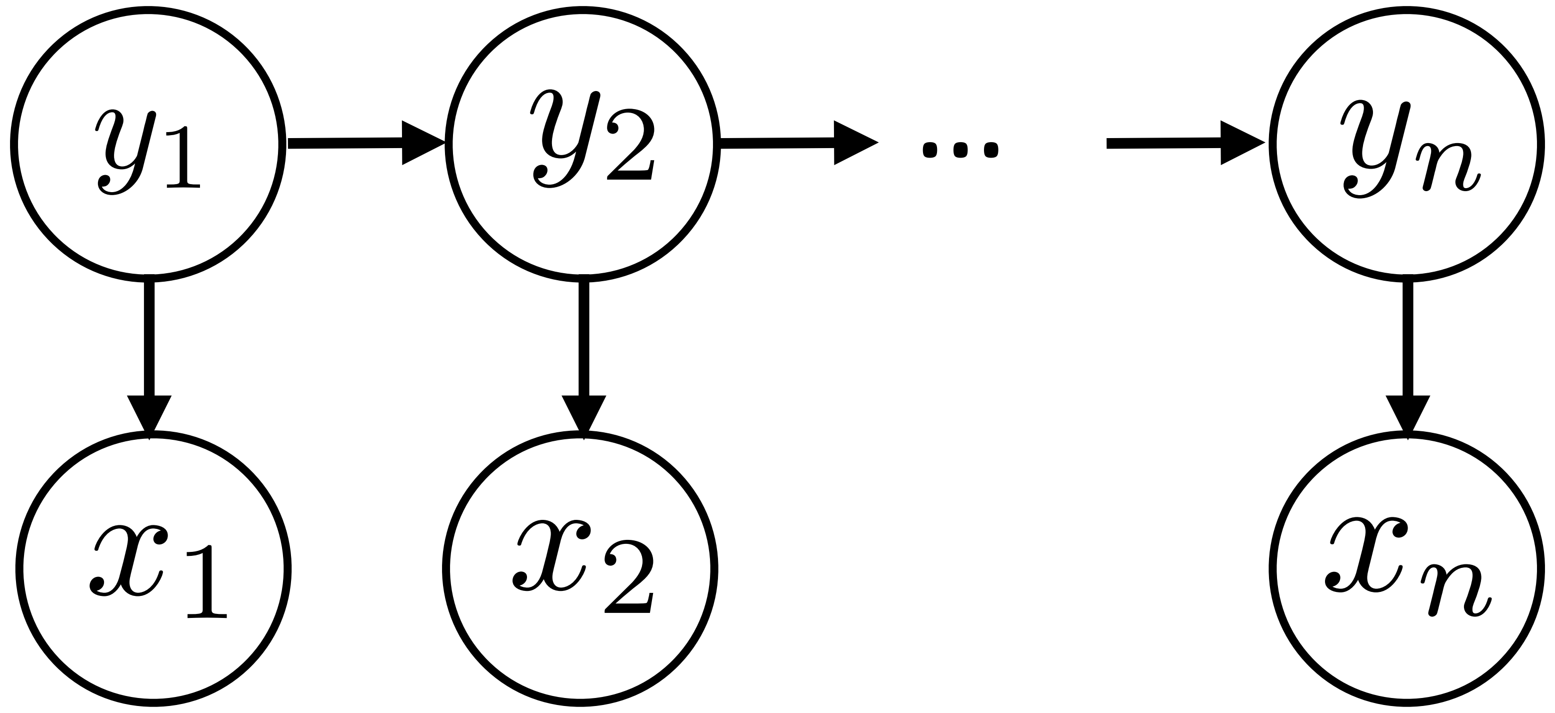
Training HMMs

- Transitions
  - Count up all pairs (y_i, y_{i+1}) in the training data
  - Count up occurrences of what tag *T* can transition to
  - Normalize to get a distribution for P(next tag|*T*)
  - Need to *smooth* this distribution, won't discuss here
- Emissions: similar count + normalize scheme, but trickier smoothing!
- You can write down the log likelihood and it is exactly optimized by this count + normalize scheme, so no need for SGD!

Inference: Viterbi Algorithm

Inference in HMMs

- Input $\mathbf{x} = (x_1, ..., x_n)$ Output $\mathbf{y} = (y_1, ..., y_n)$



$$P(\mathbf{y}, \mathbf{x}) = P(y_1) \prod_{i=2}^{n} \boxed{P(y_i|y_{i-1})} \prod_{i=1}^{n} P(x_i|y_i)$$

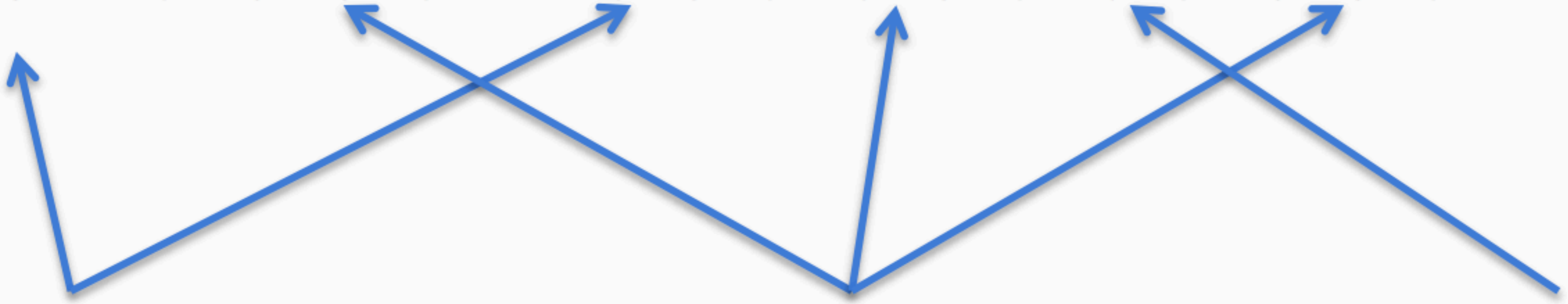
- Inference problem: $\text{argmax}_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \text{argmax}_{\mathbf{y}} \frac{P(\mathbf{y}, \mathbf{x})}{\cancel{P(\mathbf{x})}}$

- Exponentially many possible ***y*** here!

- Solution: dynamic programming (possible because of Markov structure!)

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

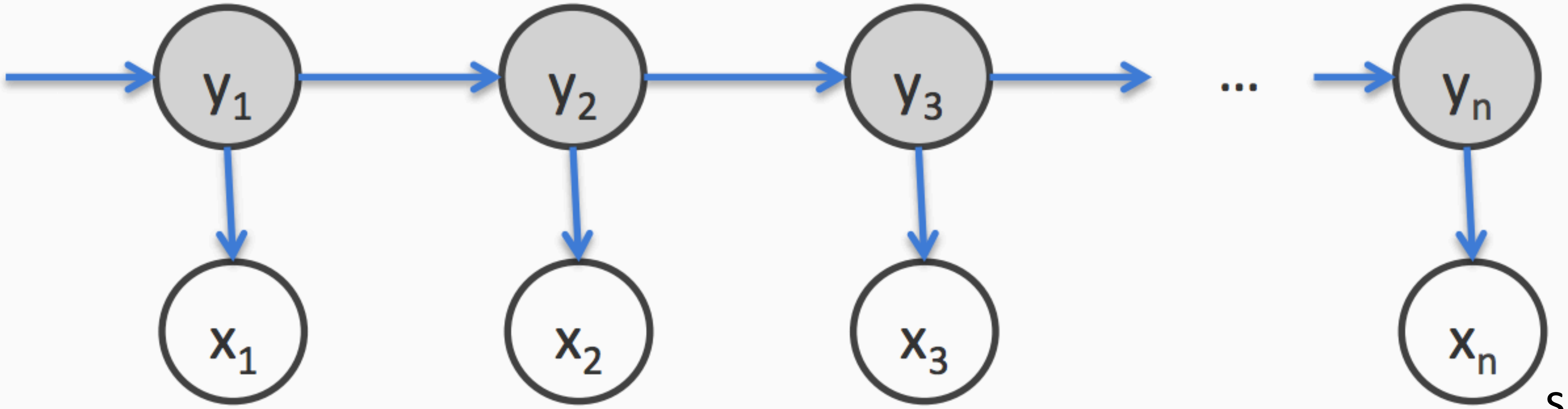
$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$



Transition probabilities

Emission probabilities

Initial probability



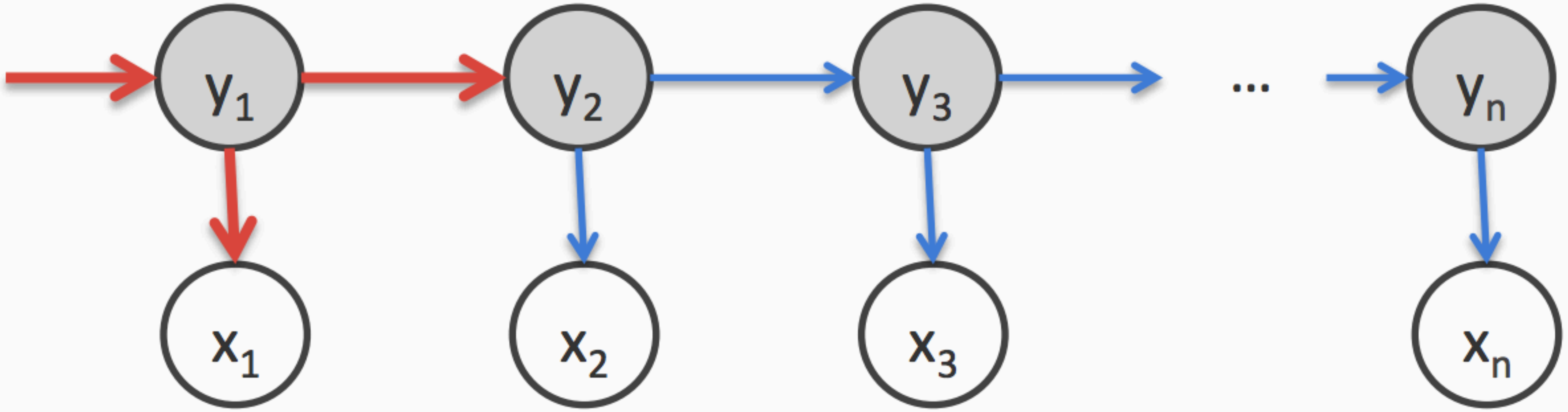
slide credit: Vivek Srikumar

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \boxed{\max_{y_1}} P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

The only terms that depend on y_1



slide credit: Vivek Srikumar

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

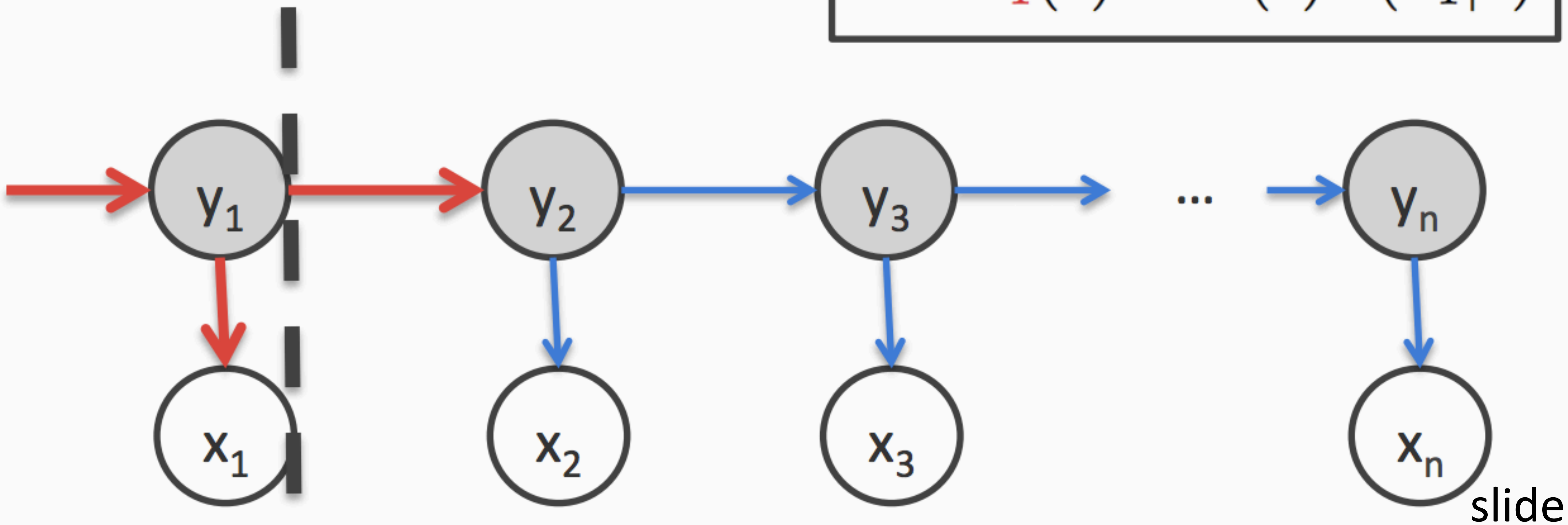
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) \mathrm{score}_1(y_1)$$

- Best (partial) score for a sequence ending in state *s*

$$\boxed{\mathrm{score}_1(s) = P(s) P(x_1|s)}$$

Abstract away the score for all decisions till here into score



slide credit: Vivek Srikumar

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

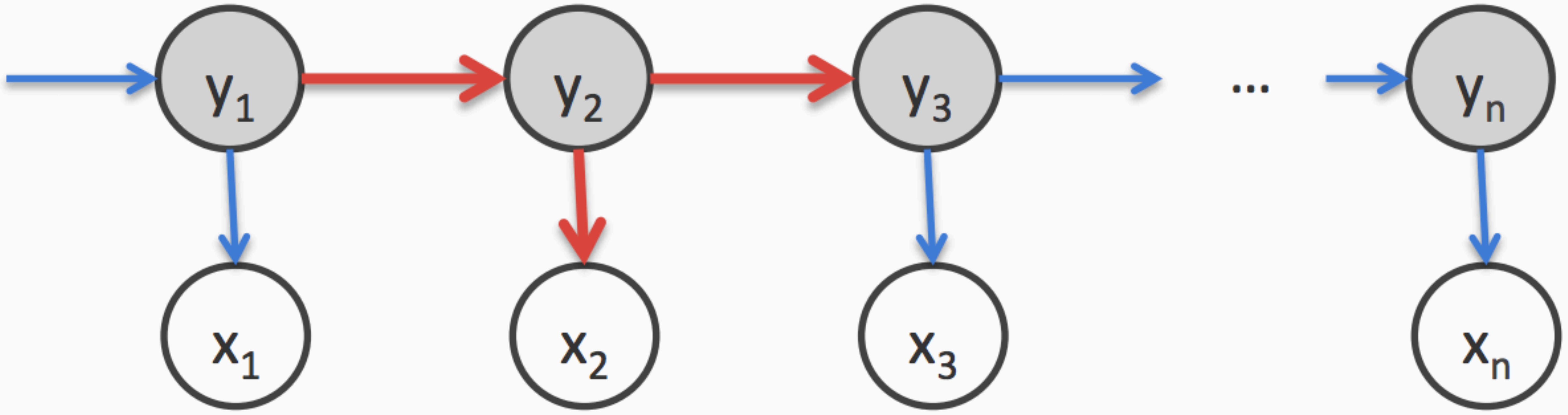
$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)$$

Only terms that depend on y_2

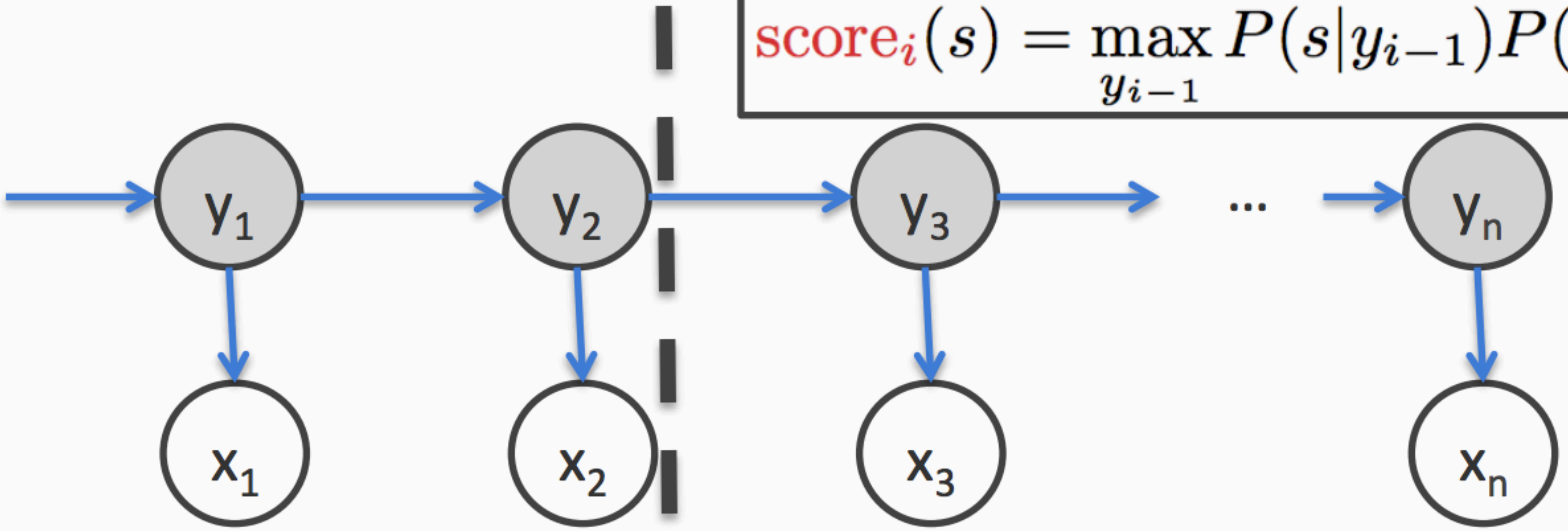


slide credit: Vivek Srikumar

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1)\prod_{i=1}^{n-1} P(y_{i+1}|y_i)\prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)$$
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\mathrm{score}_1(y_1)$$
$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3) \max_{y_1} P(y_2|y_1)P(x_2|y_2)\mathrm{score}_1(y_1)$$
$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_2} P(y_3|y_2)P(x_3|y_3)\mathrm{score}_2(y_2)$$

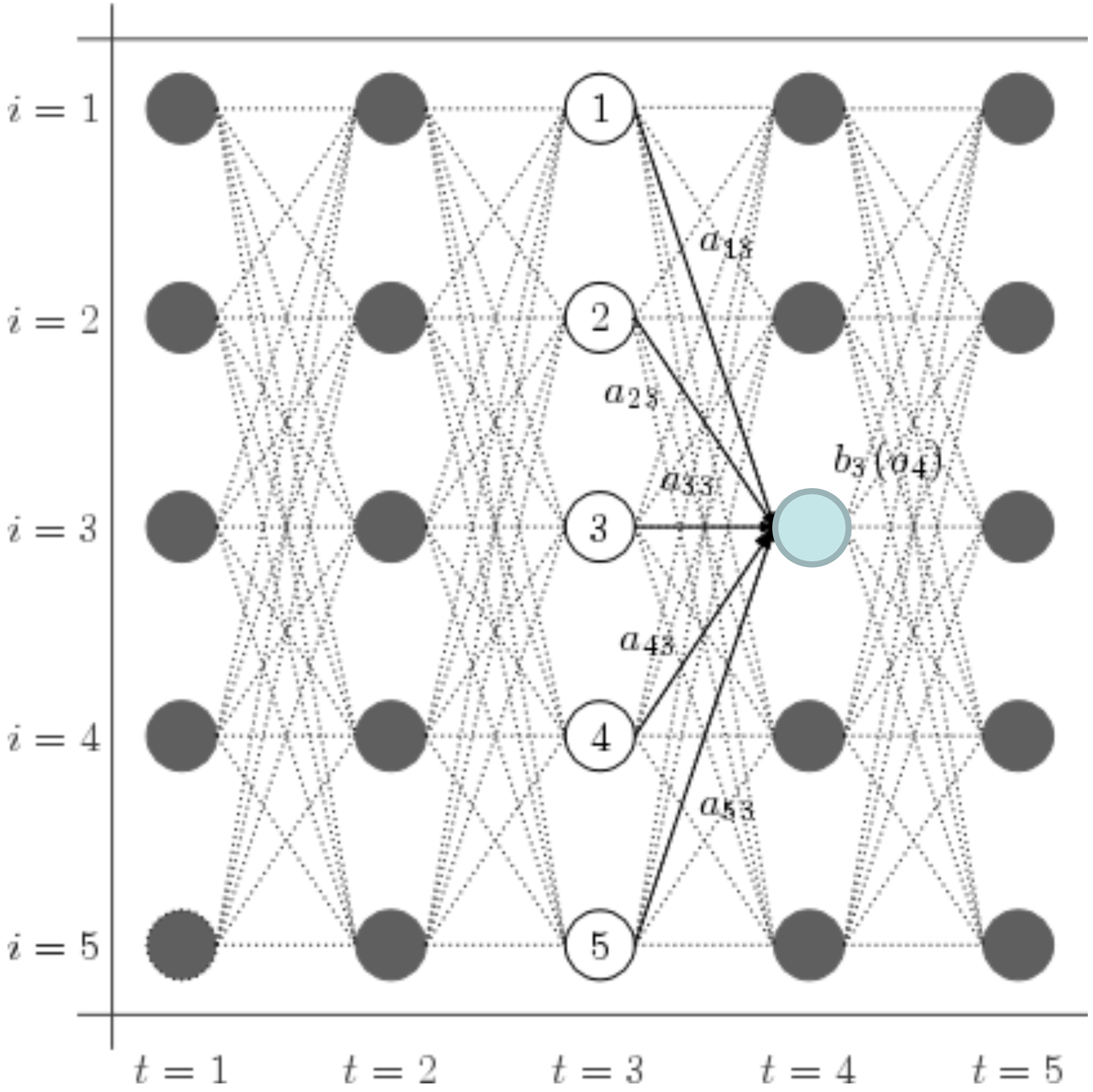
$$\boxed{\mathrm{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\mathrm{score}_{i-1}(y_{i-1})}$$



Abstract away the score for all decisions till here into score

slide credit: Vivek Srikumar

Viterbi Algorithm



- “Think about” all possible immediate prior state values. Everything before that has already been accounted for by earlier stages.

slide credit: Dan Klein

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

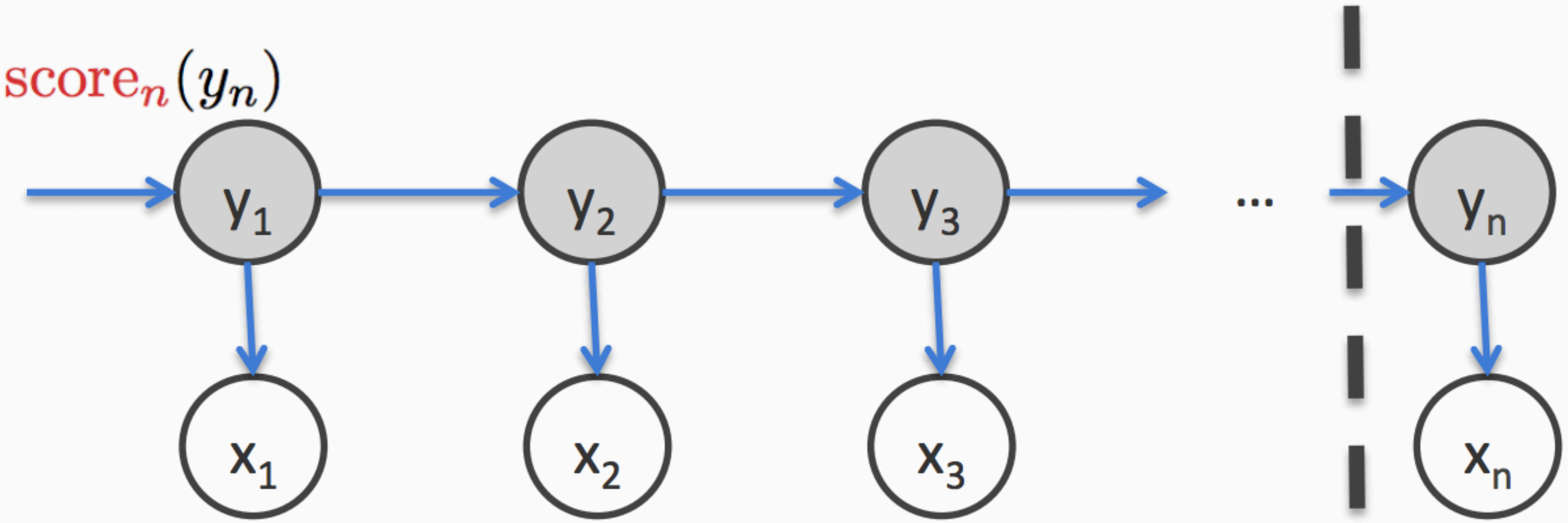
$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) \text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2) P(x_3|y_3) \max_{y_1} P(y_2|y_1) P(x_2|y_2) \text{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2) P(x_3|y_3) \text{score}_2(y_2)$$

$$\vdots$$

$$= \max_{y_n} \text{score}_n(y_n)$$



Abstract away the score for all decisions till here into score

slide credit: Vivek Srikumar

$$P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)$$

$$\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)$$

$$= \max_{y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1) P(x_2|y_2) \mathrm{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2) P(x_3|y_3) \max_{y_1} P(y_2|y_1) P(x_2|y_2) \mathrm{score}_1(y_1)$$

$$= \max_{y_3, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots \max_{y_2} P(y_3|y_2) P(x_3|y_3) \mathrm{score}_2(y_2)$$

$$\vdots$$

$$= \max_{y_n} \mathrm{score}_n(y_n)$$

$$\mathrm{score}_1(s) = P(s) P(x_1|s)$$

$$\mathrm{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1}) P(x_i|s) \mathrm{score}_{i-1}(y_{i-1})$$

slide credit: Vivek Srikumar

1. Initial: For each state s, calculate

$$\text{score}_1(s) = P(s)P(x_1|s) = \pi_s B_{x_1,s}$$

2. Recurrence: For i = 2 to n, for every state s, calculate

$$\text{score}_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)\text{score}_{i-1}(y_{i-1})$$

$$= \max_{y_{i-1}} A_{y_{i-1},s} B_{s,x_i} \text{score}_{i-1}(y_{i-1})$$

3. Final state: calculate

$$\max_{\mathbf{y}} P(\mathbf{y},\mathbf{x}|\pi,A,B) = \max_{s} \text{score}_n(s)$$

π: Initial probabilities
A: Transitions
B: Emissions

This only calculates the max. To get final answer (*argmax*),

- keep track of which state corresponds to the max at each step
- build the answer using these back pointers

slide credit: Vivek Srikumar

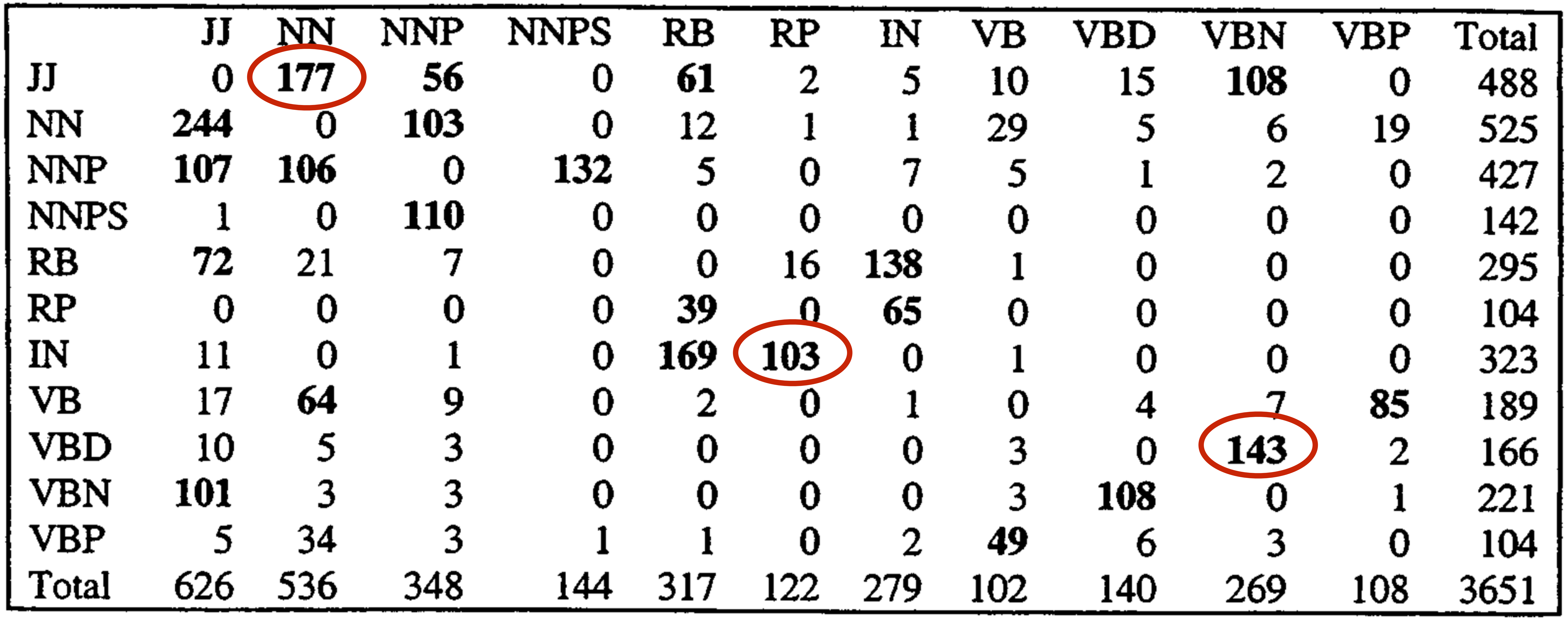
POS Taggers

HMM POS Tagging

- Penn Treebank English POS tagging: 44 tags
- Baseline: assign each word its most frequent tag: ~90% accuracy
- Trigram HMM (states are *pairs* of tags): ~95% accuracy / 55% on words not seen in train
- TnT tagger (Brants 1998, tuned HMM): 96.2% acc / 86.0% on unks
- CRF tagger (Toutanova + Manning 2000): 96.9% / 87.0%
- State-of-the-art (BiLSTM-CRFs, BERT): 97.5% / 89%+

Slide credit: Dan Klein

Errors



	JJ	NN	NNP	NNPS	RB	RP	IN	VB	VBD	VBN	VBP	Total
JJ	0	**177**	**56**	0	**61**	2	5	10	15	**108**	0	488
NN	**244**	0	**103**	0	12	1	1	29	5	6	19	525
NNP	**107**	**106**	0	**132**	5	0	7	5	1	2	0	427
NNPS	1	0	**110**	0	0	0	0	0	0	0	0	142
RB	**72**	21	7	0	0	16	**138**	1	0	0	0	295
RP	0	0	0	0	**39**	0	**65**	0	0	0	0	104
IN	11	0	1	0	**169**	**103**	0	1	0	0	0	323
VB	17	**64**	9	0	2	0	1	0	4	7	**85**	189
VBD	10	5	3	0	0	0	0	3	0	**143**	2	166
VBN	**101**	3	3	0	0	0	0	3	**108**	0	1	221
VBP	5	34	3	1	1	0	2	**49**	6	3	0	104
Total	626	536	348	144	317	122	279	102	140	269	108	3651

JJ/NN NN
official knowledge

VBD RP/IN DT NN
made up the story

RB VBD/VBN NNS
recently sold shares

(NN NN: tax cut, art gallery, …)

Slide credit: Dan Klein / Toutanova + Manning (2000)

Remaining Errors

- Lexicon gap (word not seen with that tag in training) 4.5%
- Unknown word: 4.5%
- Could get right: 16% (many of these involve parsing!)
- Difficult linguistics: 20%

VBD / VBP? (past or present?)

They set up absurd situations, detached from reality

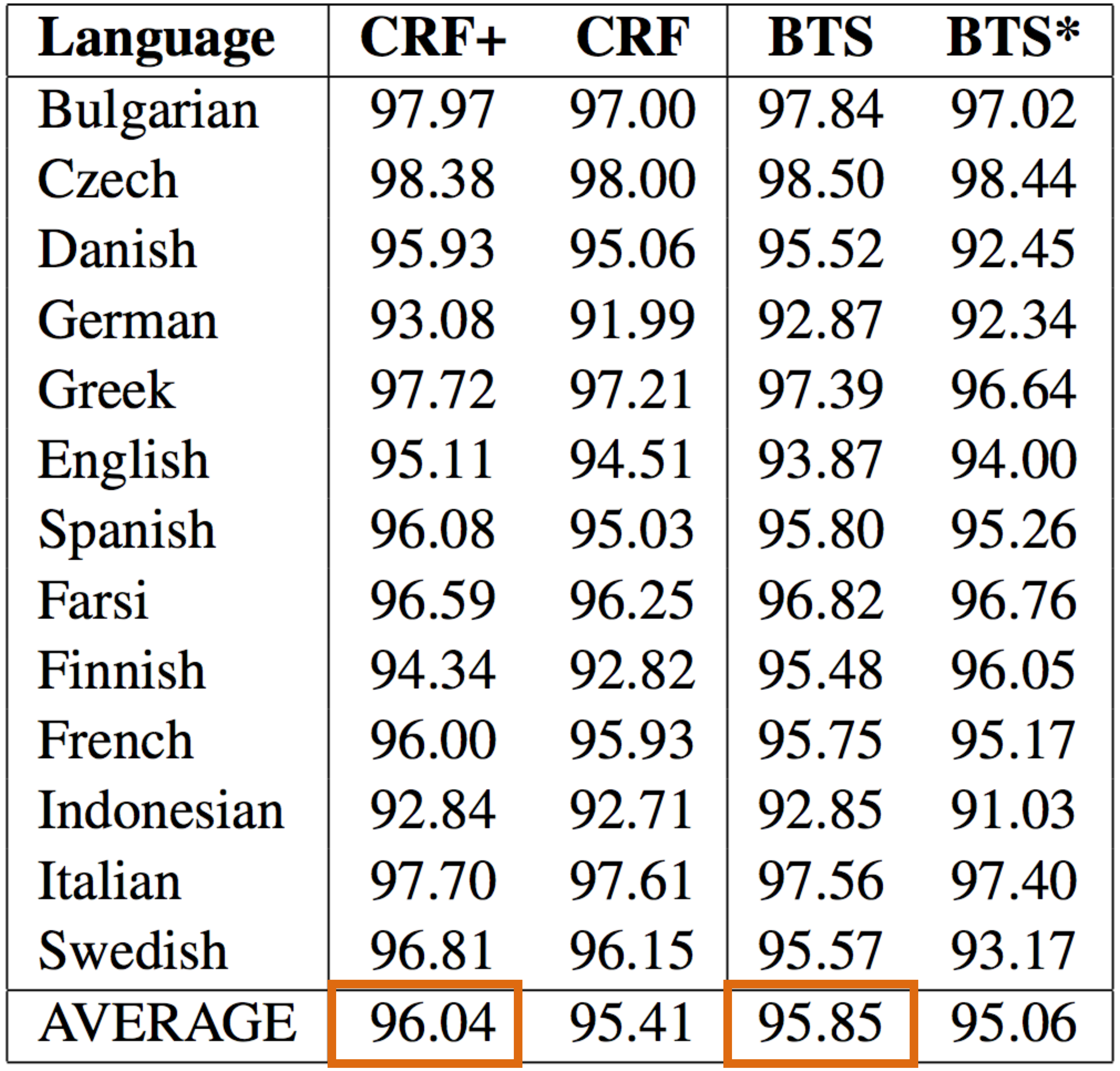
- Underspecified / unclear, gold standard inconsistent / wrong: **58%**

adjective or verbal participle? JJ / VBN?

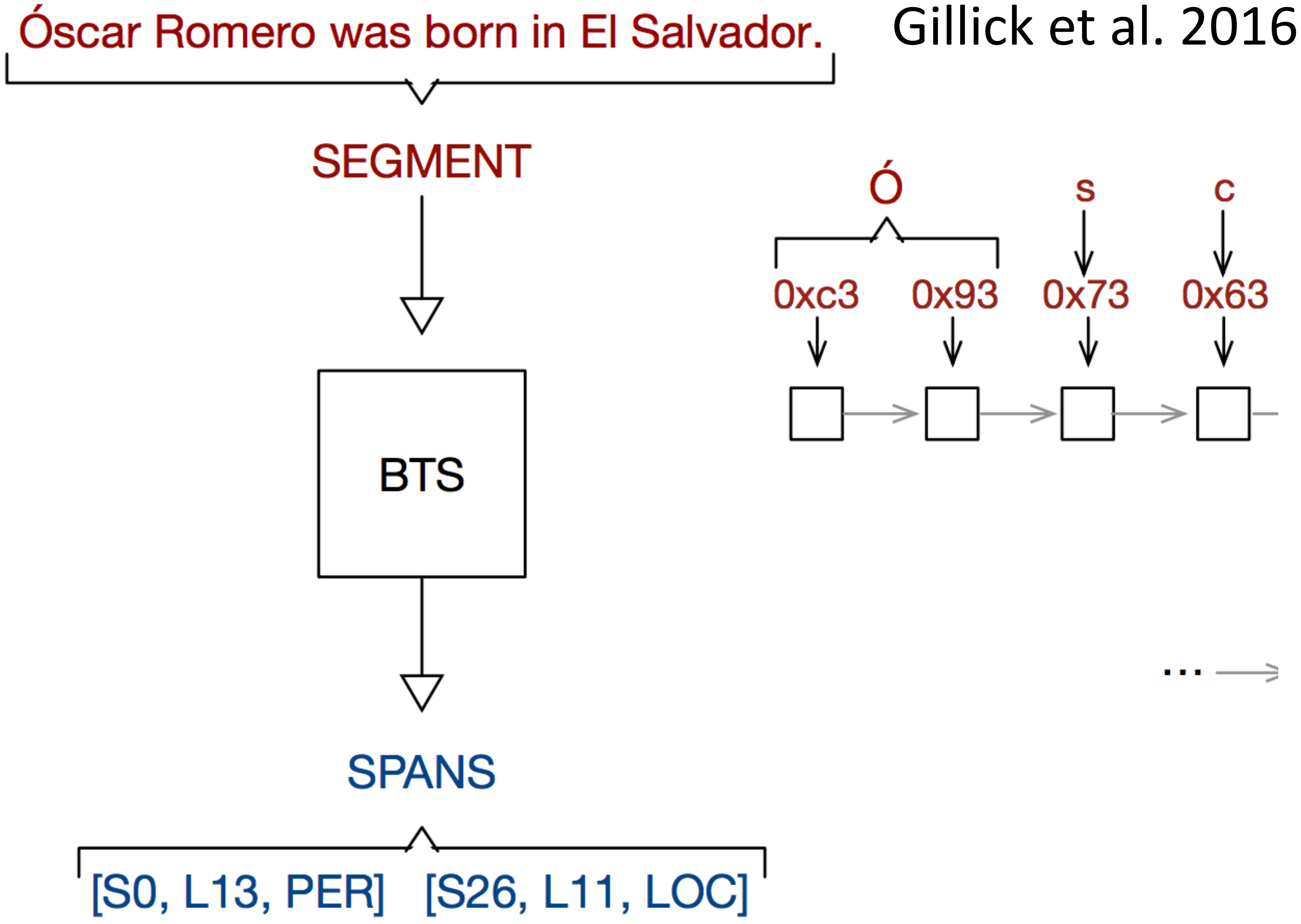
a $ 10 million fourth-quarter charge against discontinued operations

Manning 2011 “Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?”

Other Languages



Language	CRF+	CRF	BTS	BTS*
Bulgarian	97.97	97.00	97.84	97.02
Czech	98.38	98.00	98.50	98.44
Danish	95.93	95.06	95.52	92.45
German	93.08	91.99	92.87	92.34
Greek	97.72	97.21	97.39	96.64
English	95.11	94.51	93.87	94.00
Spanish	96.08	95.03	95.80	95.26
Farsi	96.59	96.25	96.82	96.76
Finnish	94.34	92.82	95.48	96.05
French	96.00	95.93	95.75	95.17
Indonesian	92.84	92.71	92.85	91.03
Italian	97.70	97.61	97.56	97.40
Swedish	96.81	96.15	95.57	93.17
AVERAGE	96.04	95.41	95.85	95.06



Gillick et al. 2016

- Universal POS tagset (~12 tags), cross-lingual model works as well as tuned CRF using external resources

NER

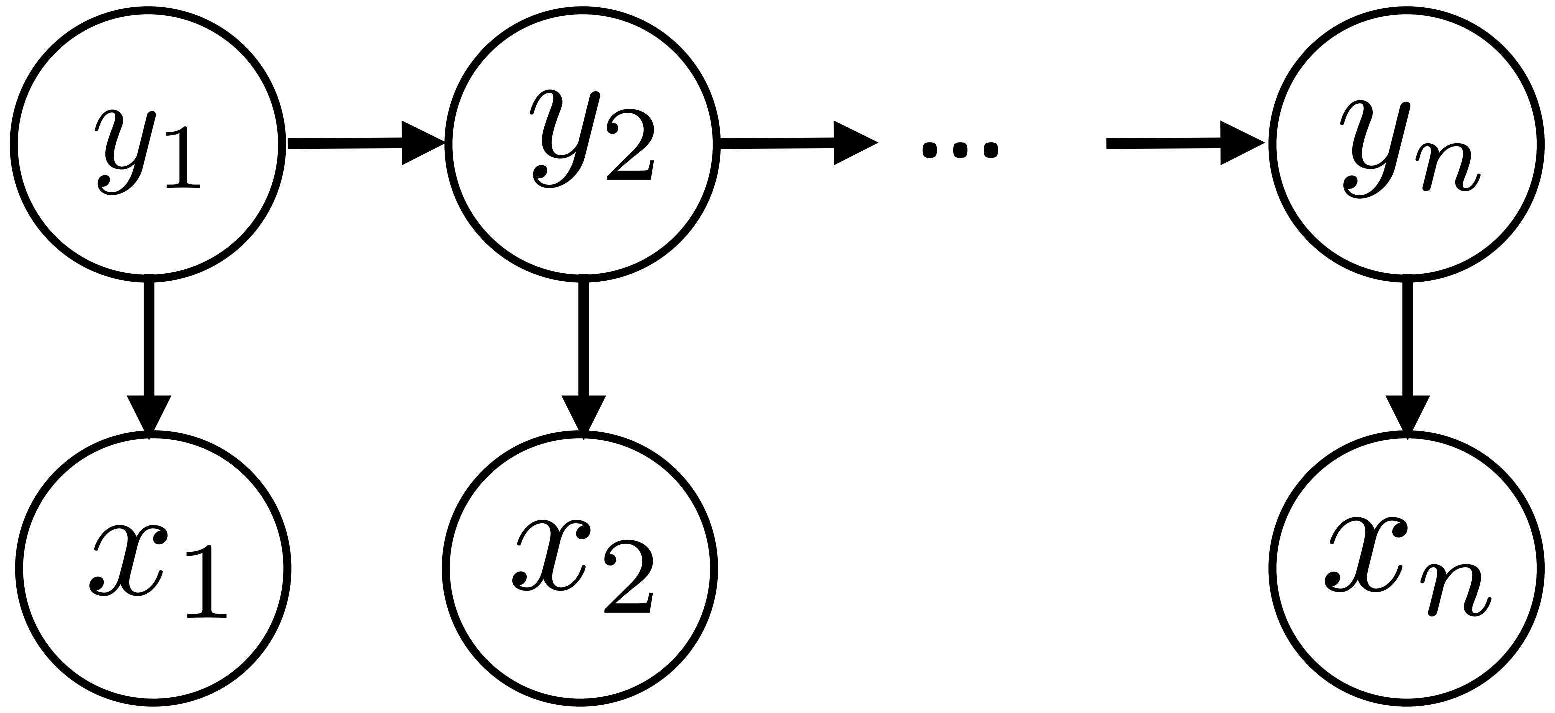
Named Entity Recognition

B-PER	I-PER	O	O	O	B-LOC	O	O	O	B-ORG	O	O
Barack	*Obama*	*will*	*travel*	*to*	*Hangzhou*	*today*	*for*	*the*	*G20*	*meeting*	*.*
PERSON					LOC				ORG		

- BIO tagset: begin, inside, outside
- Sequence of tags — should we use an HMM?
- Why might an HMM not do so well here?
  - Lots of O's
  - Insufficient features/capacity with multinomials (especially for unks)

HMMs Pros and Cons

- Big advantage: transitions, scoring pairs of adjacent y's



- Big downside: not able to incorporate useful word context information

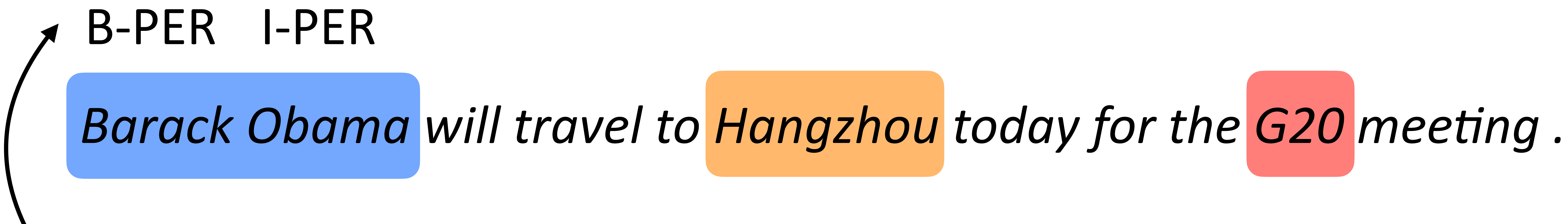
- Solution: switch from generative to discriminative model (conditional random fields) so we can condition on the *entire input*.

- Conditional random fields: logistic regression + features on pairs of y's

Conditional Random Fields

Conditional Random Fields

- Flexible discriminative model for tagging tasks that can use arbitrary features of the input. Similar to logistic regression, but *structured*



B-PER I-PER

Barack Obama will travel to Hangzhou today for the G20 meeting .

Curr_word=Barack & **Label=B-PER**

Next_word=Obama & **Label=B-PER**

Curr_word_starts_with_capital=True & **Label=B-PER**

Posn_in_sentence=1st & **Label=B-PER**

Label=B-PER & Next-Label = I-PER

…

Tagging with Logistic Regression

- Logistic regression over each tag individually:

"different features" approach to features for a single tag

$$P(y_i = y|\mathbf{x}, i) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(y, i, \mathbf{x}))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \mathbf{f}(y', i, \mathbf{x}))}$$

Probability of the *i*th word getting assigned tag *y* (B-PER, etc.)

Tagging with Logistic Regression

- Logistic regression over each tag individually:

“different features” approach to features for a single tag

$$P(y_i = y|\mathbf{x}, i) = \frac{\exp(\mathbf{w}^\top \mathbf{f}(y, i, \mathbf{x}))}{\sum_{y' \in \mathcal{Y}} \exp(\mathbf{w}^\top \mathbf{f}(y', i, \mathbf{x}))}$$

- Over all tags:

$$P(\mathbf{y} = \tilde{\mathbf{y}}|\mathbf{x}) = \prod_{i=1}^{n} P(y_i = \tilde{y}_i|\mathbf{x}, i) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{f}(\tilde{y}_i, i, \mathbf{x})\right)$$

- Score of a prediction: sum of weights dot features over each individual predicted tag (this is a simple CRF but not the general form)
- Set *Z* equal to the product of denominators
- Conditional model: ***x*** is observed, unlike in HMMs

Example: “Emission Features” f_e

B-PER I-PER O O
Barack Obama will travel

feats = $\mathbf{f}_e$(B-PER, i=1, $\mathbf{x}$) + $\mathbf{f}_e$(I-PER, i=2, $\mathbf{x}$) + $\mathbf{f}_e$(O, i=3, $\mathbf{x}$) + $\mathbf{f}_e$(O, i=4, $\mathbf{x}$)

[CurrWord=*Obama* & label=I-PER, PrevWord=*Barack* & label=I-PER, CurrWordIsCapitalized & label=I-PER, …]

B-PER B-PER O O
Barack Obama will travel

feats = $\mathbf{f}_e$(B-PER, i=1, $\mathbf{x}$) + $\mathbf{f}_e$(B-PER, i=2, $\mathbf{x}$) + $\mathbf{f}_e$(O, i=3, $\mathbf{x}$) + $\mathbf{f}_e$(O, i=4, $\mathbf{x}$)

Adding Structure

$$P(\mathbf{y} = \tilde{\mathbf{y}}|\mathbf{x}) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{f}(\tilde{y}_i, i, \mathbf{x})\right)$$

- We want to be able to learn that some tags don't follow other tags — want to have features on tag *pairs*

$$P(\mathbf{y} = \tilde{\mathbf{y}}|\mathbf{x}) = \frac{1}{Z} \exp\left(\sum_{i=1}^{n} \mathbf{w}^\top \mathbf{f}_e(\tilde{y}_i, i, \mathbf{x}) + \sum_{i=2}^{n} \mathbf{w}^\top \mathbf{f}_t(\tilde{y}_{i-1}, \tilde{y}_i, i, \mathbf{x})\right)$$

- Score: sum of weights dot $\mathbf{f}_e$ features over each predicted tag ("emissions") plus sum of weights dot $\mathbf{f}_t$ features over tag pairs ("transitions")

- This is a sequential CRF

Example

B-PER I-PER O O
Barack Obama will travel

feats = $\mathbf{f}_e(\text{B-PER}, i=1, \mathbf{x}) + \mathbf{f}_e(\text{I-PER}, i=2, \mathbf{x}) + \mathbf{f}_e(\text{O}, i=3, \mathbf{x}) + \mathbf{f}_e(\text{O}, i=4, \mathbf{x})$
$+ \mathbf{f}_t(\text{B-PER}, \text{I-PER}, i=1, \mathbf{x}) + \mathbf{f}_t(\text{I-PER}, \text{O}, i=2, \mathbf{x}) + \mathbf{f}_t(\text{O}, \text{O}, i=3, \mathbf{x})$

B-PER B-PER O O
Barack Obama will travel

feats = $\mathbf{f}_e(\text{B-PER}, i=1, \mathbf{x}) + \mathbf{f}_e(\text{B-PER}, i=2, \mathbf{x}) + \mathbf{f}_e(\text{O}, i=3, \mathbf{x}) + \mathbf{f}_e(\text{O}, i=4, \mathbf{x})$
$+ \mathbf{f}_t(\text{B-PER}, \text{B-PER}, i=1, \mathbf{x}) + \mathbf{f}_t(\text{B-PER}, \text{O}, i=2, \mathbf{x}) + \mathbf{f}_t(\text{O}, \text{O}, i=3, \mathbf{x})$

- *Obama* can start a new named entity (emission feats look okay), but we’re not likely to have two PER entities in a row (transition feats)

Sequential CRFs

$$P(\mathbf{y} = \tilde{\mathbf{y}}|\mathbf{x}) = \frac{1}{Z}\exp\left(\sum_{i=1}^{n}\mathbf{w}^\top\mathbf{f}_e(\tilde{y}_i, i, \mathbf{x}) + \sum_{i=2}^{n}\mathbf{w}^\top\mathbf{f}_t(\tilde{y}_{i-1}, \tilde{y}_i, i, \mathbf{x})\right)$$

- Critical property: this structure is allows us to use dynamic programming (Viterbi) to sum or max over all sequences

- **Inference:** use Viterbi, just replace probabilities with exponentiated weights * features

- **Learning:** need another dynamic program (forward-backward) to compute gradients

CRFs Today

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} \prod_{i=2}^{n} \exp(\phi_t(y_{i-1}, y_i)) \prod_{i=1}^{n} \exp(\phi_e(y_i, i, \mathbf{x}))$$

- Generalization of sequential CRF with arbitrary function phi.
  We can replace these with computations from neural nets (e.g., contextualized embedding from BERT -> linear layer to produce phi)

- Can backpropagate into BERT

- “Neural CRFs” for tagging (Lample et al., 2016), parsing (Durrett and Klein, 2015; Dozat and Manning, 2016)

CRFs Today

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z}\prod_{i=2}^{n}\exp(\phi_t(y_{i-1}, y_i))\prod_{i=1}^{n}\exp(\phi_e(y_i, i, \mathbf{x}))$$

- Why aren’t CRFs used more today?
  - We don’t often need to **score** transitions: If you have hard constraints (e.g., cannot follow B-PER with I-ORG), you can simply integrate these into inference. Train BERT to predict each label individually, then use Viterbi to get a coherent sequence.
  - ChatGPT and other such systems are decent at learning structural constraints — so bigger models also learn most of the constraints you really want

Takeaways

- POS and NER are two ways of capturing sequential structures
  - POS: syntax, each word has a tag
  - NER: spans, but we can turn them into tags with BIO
- Can handle these with generative or discriminative models, but CRFs are most typically used (although these days you can also just ask ChatGPT…)
- Next time: move from sequences to trees