COMP 786 (Fall 2020)
Natural Language Processing

Lecture 3: POS-Tagging, NER, Seq Labeling, Coreference

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

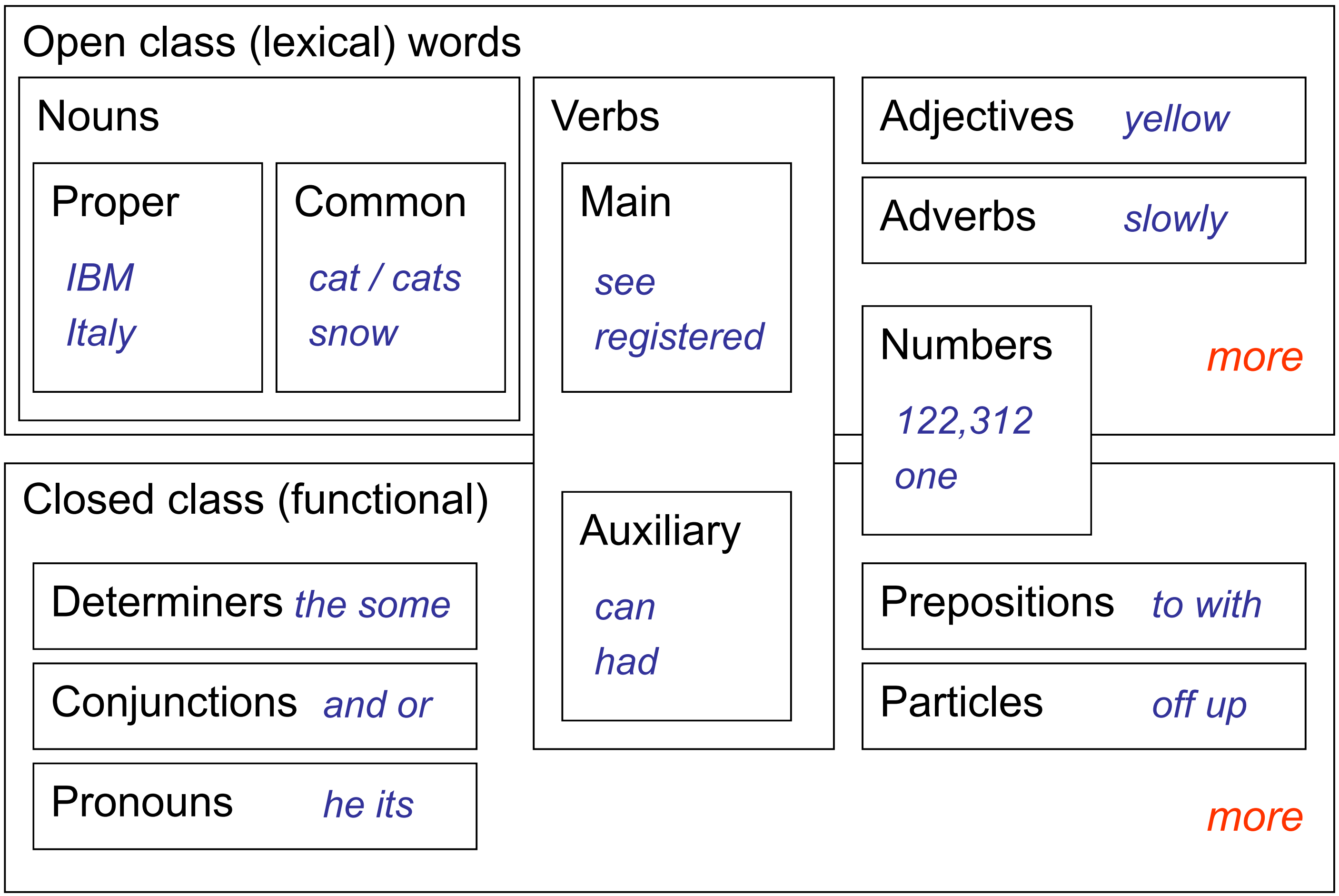
Mohit Bansal

(various slides adapted/borrowed from courses by Dan Klein, Richard Socher, Chris Manning, JurafskyMartin-SLP3, others)

Part-of-Speech Tagging

Part-of-Speech Tagging

- Basic form of linguistic structure: ‘syntactic word classes’
- Tag sequence of words w/ syntactic categories (noun, verb, prep, etc.)



Penn Treebank Tagset

CC	conjunction, coordinating	and both but either or
CD	numeral, cardinal	mid-1890 nine-thirty 0.5 one
DT	determiner	a all an every no that the
EX	existential there	there
FW	foreign word	gemeinschaft hund ich jeux
IN	preposition or conjunction, subordinating	among whether out on by if
JJ	adjective or numeral, ordinal	third ill-mannered regrettable
JJR	adjective, comparative	braver cheaper taller
JJS	adjective, superlative	bravest cheapest tallest
MD	modal auxiliary	can may might will would
NN	noun, common, singular or mass	cabbage thermostat investment subhumanity
NNP	noun, proper, singular	Motown Cougar Yvette Liverpool
NNPS	noun, proper, plural	Americans Materials States
NNS	noun, common, plural	undergraduates bric-a-brac averages
POS	genitive marker	' 's
PRP	pronoun, personal	hers himself it we them
PRP$	pronoun, possessive	her his mine my our ours their thy your
RB	adverb	occasionally maddeningly adventurously
RBR	adverb, comparative	further gloomier heavier less-perfectly
RBS	adverb, superlative	best biggest nearest worst
RP	particle	aboard away back by on open through
TO	"to" as preposition or infinitive marker	to
UH	interjection	huh howdy uh whammo shucks heck
VB	verb, base form	ask bring fire see take
VBD	verb, past tense	pleaded swiped registered saw
VBG	verb, present participle or gerund	stirring focusing approaching erasing
VBN	verb, past participle	dilapidated imitated reunifed unsettled
VBP	verb, present tense, not 3rd person singular	twist appear comprise mold postpone
VBZ	verb, present tense, 3rd person singular	bases reconstructs marks uses
WDT	WH-determiner	that what whatever which whichever
WP	WH-pronoun	that what whatever which who whom
WP$	WH-pronoun, possessive	whose
WRB	Wh-adverb	however whenever where why

Part-of-Speech Ambiguities

- A word can have multiple parts of speech

VBD		VB			
VBN	VBZ	VBP	VBZ		
NNP	NNS	NN	NNS	CD	NN

Fed raises interest rates 0.5 percent

Mrs./NNP Shaefer/NNP never/RB got/VBD **around/RP** to/TO joining/VBG

All/DT we/PRP gotta/VBN do/VB is/VBZ go/VB **around/IN** the/DT corner/NN

Chateau/NNP Petrus/NNP costs/VBZ **around/RB** 250/CD

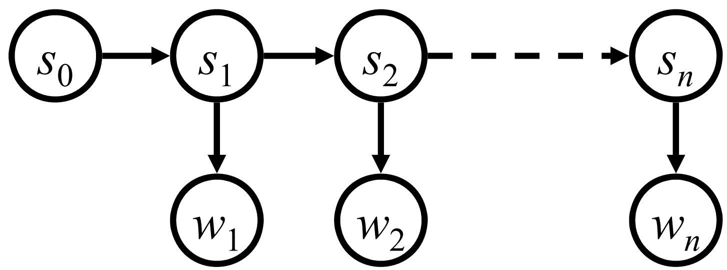
- Disambiguating features: lexical identity (word), context, morphology (suffixes, prefixes), capitalization, gazetteers (dictionaries), …

Uses of Part-of-Speech Tagging

- Useful in itself:
  - Text-to-speech: *read, lead, record*
  - Lemmatization: *saw*[v] → *see*, *saw*[n] → *saw*
  - Shallow Chunking: grep {JJ | NN}* {NN | NNS}
- Useful for downstream tasks (e.g., in parsing, and as features in various word/text classification tasks)
- Preprocessing step in parsing: allows fewer parse options if less tag ambiguity (but some cases still decided by parser)
- Demos: http://nlp.stanford.edu:8080/corenlp/

Classic Solution: HMMs

- Generative mode with state sequence and emissions at every time step:

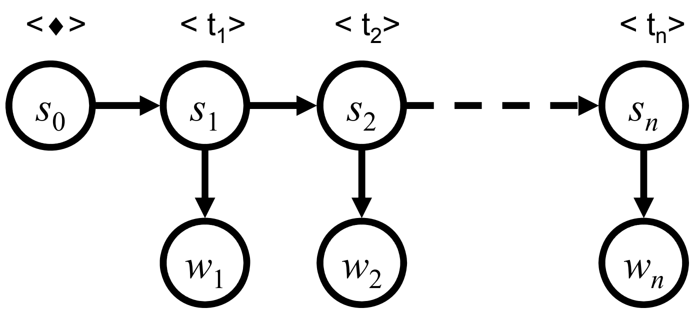


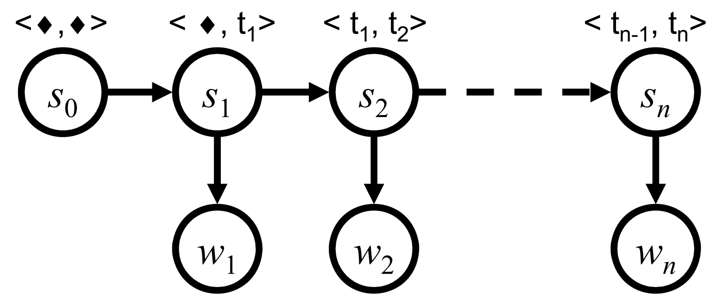
$$P(\mathbf{s}, \mathbf{w}) = \prod_i P(s_i|s_{i-1})P(w_i|s_i)$$

- Several strong independence assumptions!
  - States = POS tag n-grams
  - Next tag only depends on k previous tags
  - Word generated only depends on current tag state

States

- Markov order defines how many states in the history are being conditioned on, e.g., 1 = bigrams, 2 = trigrams





Maximum-Likelihood Estimates

The A matrix contains the tag transition probabilities $P(t_i|t_{i-1})$ which represent the probability of a tag occurring given the previous tag. For example, modal verbs like *will* are very likely to be followed by a verb in the base form, a VB, like *race*, so we expect this probability to be high. We compute the maximum likelihood estimate of this transition probability by counting, out of the times we see the first tag in a labeled corpus, how often the first tag is followed by the second:

$$P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})} \tag{8.9}$$

In the WSJ corpus, for example, MD occurs 13124 times of which it is followed by VB 10471, for an MLE estimate of

$$P(VB|MD) = \frac{C(MD, VB)}{C(MD)} = \frac{10471}{13124} = .80 \tag{8.10}$$

Maximum-Likelihood Estimates

The B emission probabilities, $P(w_i|t_i)$, represent the probability, given a tag (say MD), that it will be associated with a given word (say *will*). The MLE of the emission probability is

$$P(w_i|t_i) = \frac{C(t_i, w_i)}{C(t_i)} \tag{8.11}$$

Of the 13124 occurrences of MD in the WSJ corpus, it is associated with *will* 4046 times:

$$P(will|MD) = \frac{C(MD, will)}{C(MD)} = \frac{4046}{13124} = .31 \tag{8.12}$$

Estimating Transitions

- For higher order Markov chains, harder to estimate transition probabilities

- Therefore, can use standard language modeling style smoothing techniques like back-off or Kneser-Ney or Good-Turing

$$P(t_i \mid t_{i-1}, t_{i-2}) = \lambda_2 \hat{P}(t_i \mid t_{i-1}, t_{i-2}) + \lambda_1 \hat{P}(t_i \mid t_{i-1}) + (1 - \lambda_1 - \lambda_2) \hat{P}(t_i)$$

- More effective to have richer info encoded in the states themselves, i.e., state splitting/refinement

Estimating Emissions

$$P(\mathbf{s}, \mathbf{w}) = \prod_i P(s_i|s_{i-1}) \boxed{P(w_i|s_i)}$$

- Unknown and rare words (also unseen word-state pairs) big problem is estimating emission probabilities!

- Can use word shapes to get unknown word classes, e.g.,
  45,698.00 → $D^+, D^+. D^+$
  30-year → D^+-x^+

- Another trick: estimate P(t|w) instead and then invert!

Inference (Viterbi)

- After estimating all transition and emission probabilities, next step is to infer or decode the most-probable sequence of states (e.g., POS tags) given the sequence of observations (e.g., words)

$$\mathbf{t}^* = \arg\max_{\mathbf{t}} \mathsf{P}(\mathbf{t}|\mathbf{w})$$

Inference (Viterbi)

- Viterbi algo: Recursive dynamic program
- $v_t(j)$ cell of trellis represents prob of HMM in state j after first t observations & passing through most-prob state sequence $q_0\, q_1\, q_{2\ldots}\, q_{t-1}$

$$v_t(j) = \max_{i=1}^{N} v_{t-1}(i)\, a_{ij}\, b_j(o_t)$$

$v_{t-1}(i)$	the **previous Viterbi path probability** from the previous time step
a_{ij}	the **transition probability** from previous state q_i to current state q_j
$b_j(o_t)$	the **state observation likelihood** of the observation symbol o_t given the current state j

[JurafskyMartin-SLP3]

Inference (Viterbi)

- Compute the Viterbi probability by taking the most probable of the extensions of the paths that lead to the current cell.

function VITERBI(*observations* of len T, *state-graph* of len N) **returns** *best-path*

create a path probability matrix *viterbi[N+2,T]*
for each state s **from** 1 **to** N **do** ; initialization step
$viterbi[s,1] \leftarrow a_{0,s} * b_s(o_1)$
$backpointer[s,1] \leftarrow 0$
for each time step t **from** 2 **to** T **do** ; recursion step
for each state s **from** 1 **to** N **do**
$$viterbi[s,t] \leftarrow \max_{s'=1}^{N} viterbi[s',t-1] * a_{s',s} * b_s(o_t)$$
$$backpointer[s,t] \leftarrow \operatorname{argmax}_{s'=1}^{N} viterbi[s',t-1] * a_{s',s}$$
$$viterbi[q_F,T] \leftarrow \max_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
$$backpointer[q_F,T] \leftarrow \operatorname{argmax}_{s=1}^{N} viterbi[s,T] * a_{s,q_F}$$; termination step
return the backtrace path by following backpointers to states back in time from $backpointer[q_F,T]$

[JurafskyMartin-SLP3]

Inference (Viterbi)

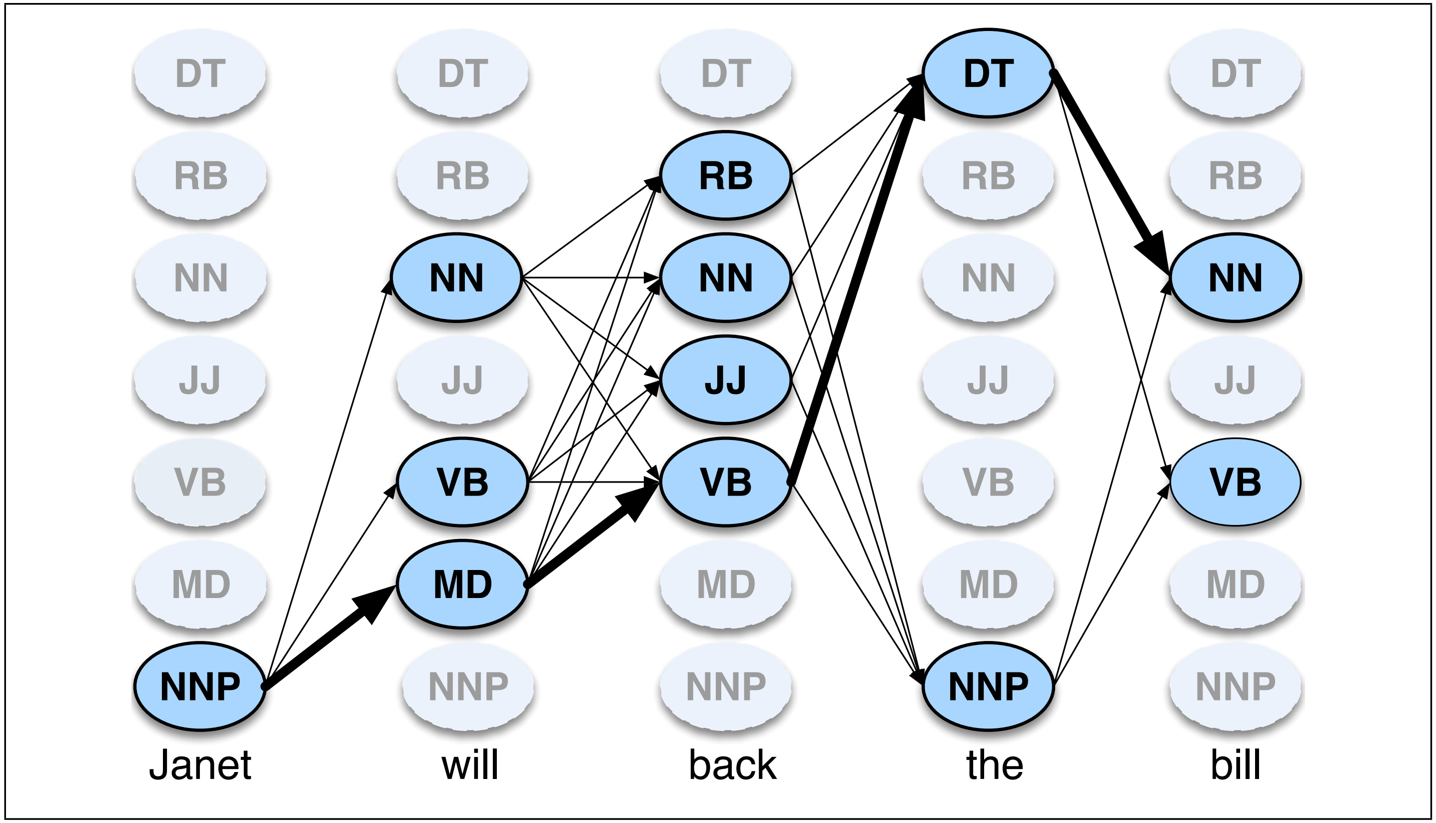
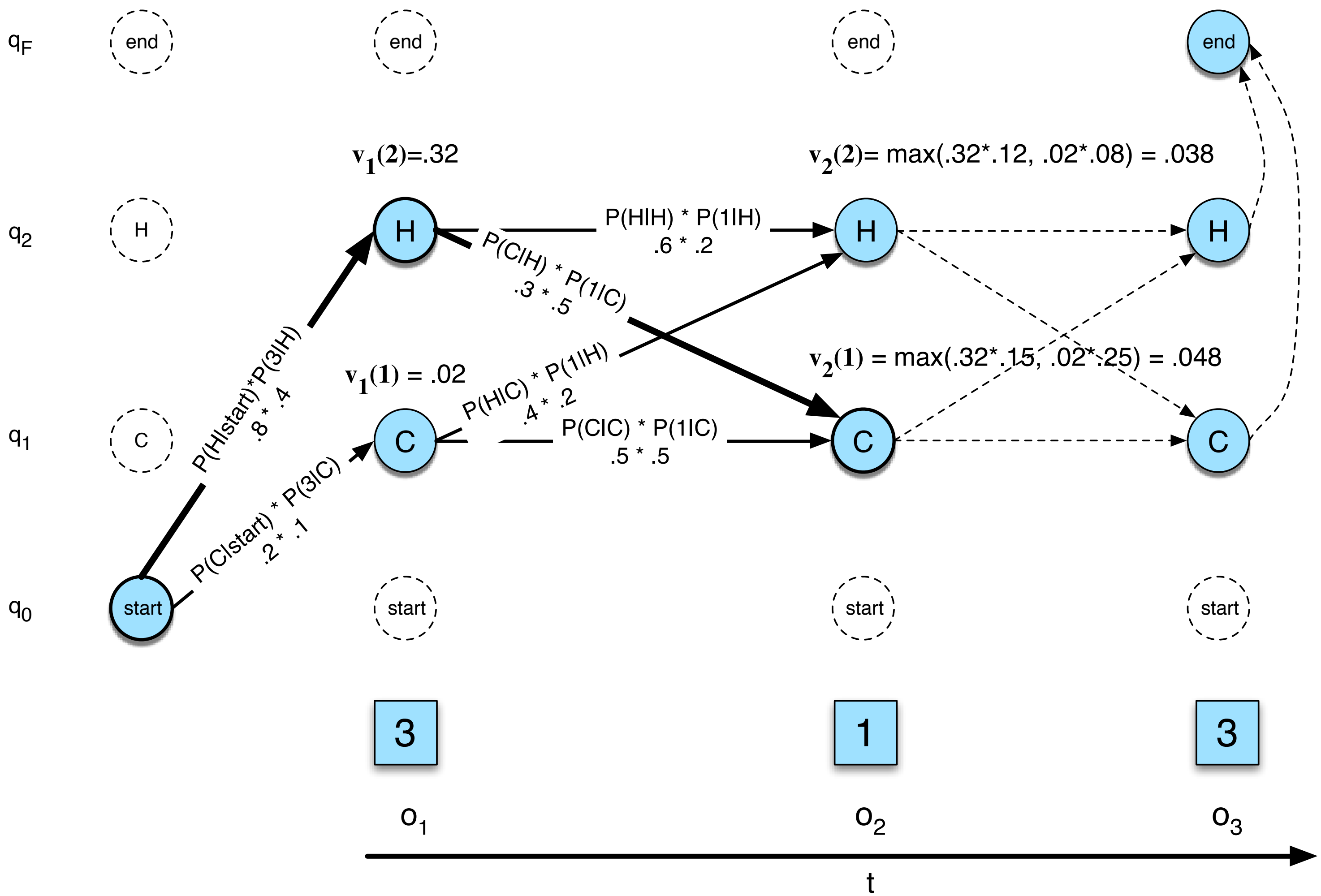


Figure 8.6 A sketch of the lattice for *Janet will back the bill*, showing the possible tags (q_i) for each word and highlighting the path corresponding to the correct tag sequence through the hidden states. States (parts of speech) which have a zero probability of generating a particular word according to the B matrix (such as the probability that a determiner DT will be realized as *Janet*) are greyed out.

[JurafskyMartin-SLP3]

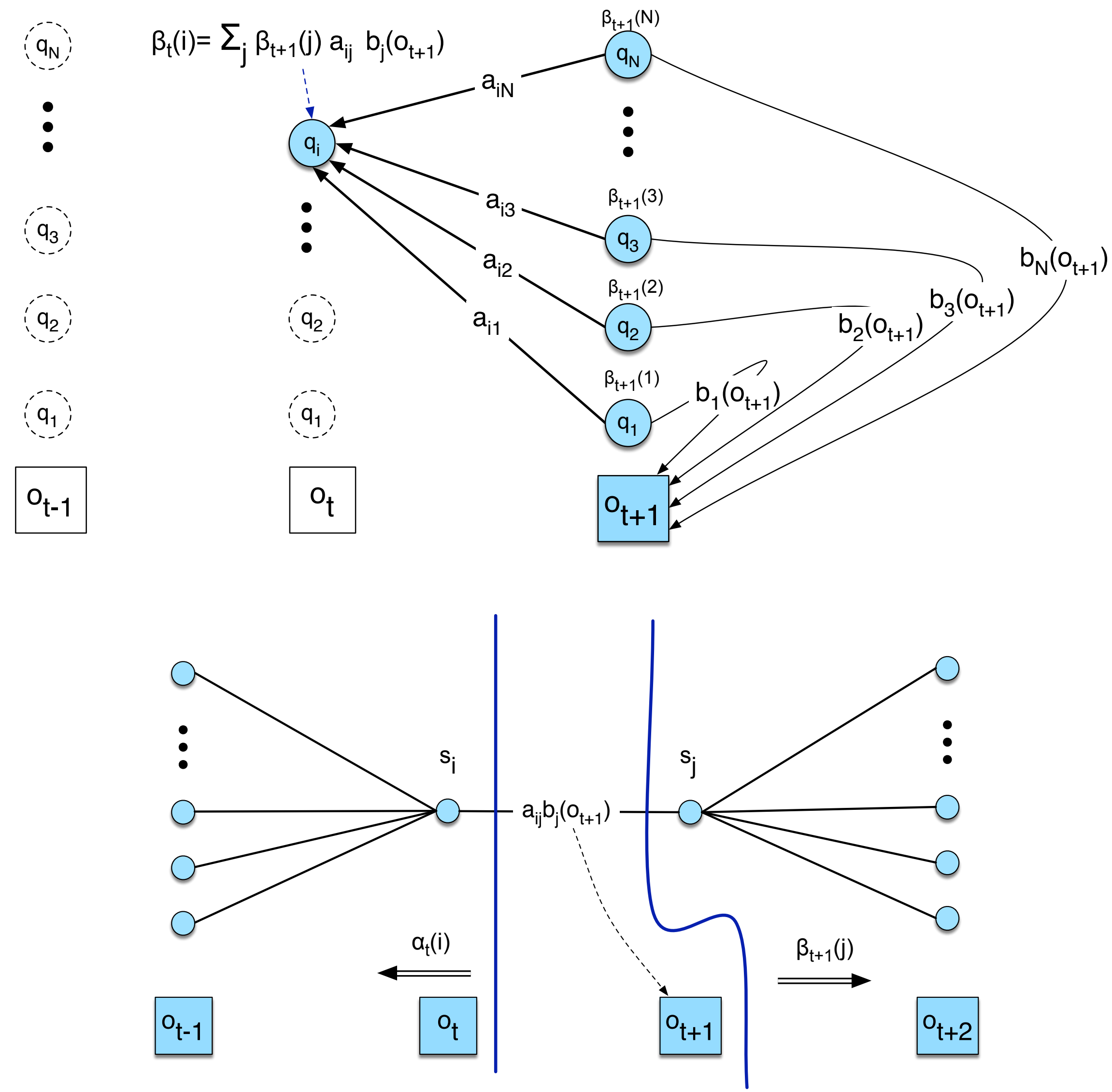
State Lattice Traversal



https://en.wikipedia.org/wiki/Viterbi_algorithm

[JurafskyMartin-SLP3]

Forward-Backward EM Algo for HMM Training



[JurafskyMartin-SLP3]

Overview of Accuracies

- Known/Unknown POS-tag accuracy history:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - TnT (HMM++): 96.2% / 86.0%
  - Maxent P(t|w): 93.7% / 82.6%
  - MEMM tagger: 96.9% / 86.9%
  - State-of-the-art: 97+% / 89+%
  - Upper bound: ~98%

Most errors on unknown words

Better Discriminative Features?

- Need richer features (both inside the word and around it)!

- Word-based feature examples:
  - Suffixes (e.g., -ly, -ing, -ed)
  - Prefixes (e.g., un-, im-, dis-)
  - Capital vs lower-cased

- Just a simple maxent tag-given-word P(t|w) feature-based model itself gets 93.7%/82.6% known/unknown POS-tagging accuracy!

Better Discriminative Features?

- Similarly, we also need linear context features, e.g., words to the right of the currently-predicted tag

```
                 RB
PRP  VBD   IN   RB  IN  PRP    VBD   .
They  left     as soon as   he    arrived .
```

- Solution: Discriminative sequence models such as CRFs and MEMMs that can incorporate such full-sentence features!

MaxEnt Markov Model (MEMM) Tagger

- Sequence model adaptation of MaxEnt (multinomial logistic regression) classifier
- MEMM = discriminative, HMM = generative
- Left-to-right local decisions, but can condition of both previous tags as well as entire input

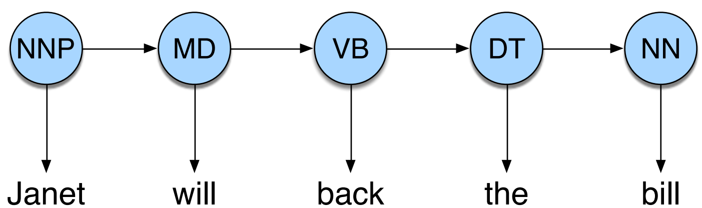
$$P(\mathbf{t}|\mathbf{w}) = \prod_i P_{\mathsf{ME}}(t_i|\mathbf{w}, t_{i-1}, t_{i-2})$$

[Ratnaparkhi, 1996]

MaxEnt Markov Model (MEMM) Tagger

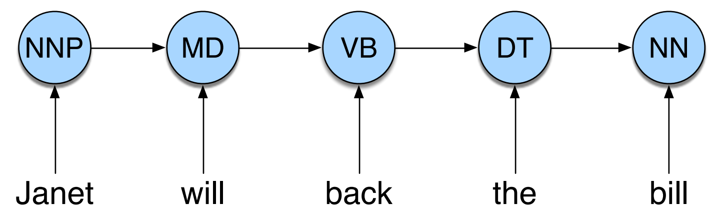
- Difference between HMM and MEMM:

HMM



$$\begin{aligned}\hat{T} &= \underset{T}{\operatorname{argmax}} P(T|W) \\ &= \underset{T}{\operatorname{argmax}} P(W|T)P(T) \\ &= \underset{T}{\operatorname{argmax}} \prod_i P(word_i|tag_i) \prod_i P(tag_i|tag_{i-1})\end{aligned}$$

MEMM

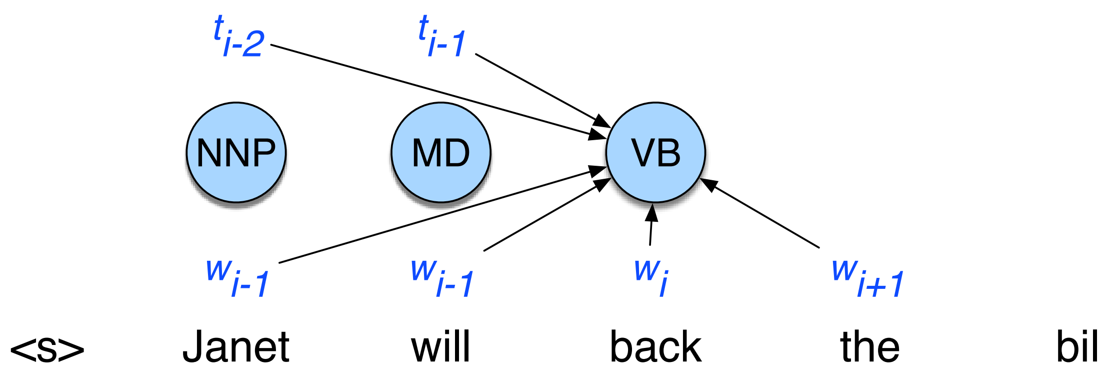


$$\begin{aligned}\hat{T} &= \underset{T}{\operatorname{argmax}} P(T|W) \\ &= \underset{T}{\operatorname{argmax}} \prod_i P(t_i|w_i, t_{i-1})\end{aligned}$$

[JurafskyMartin-SLP3]

MEMM Features

- MEMM can condition on several richer features, e.g., from words in entire input sentence



- Word shapes, tag-word n-gram templates, etc.

[JurafskyMartin-SLP3]

Perceptron Tagger

- For log-linear models, score of tags-given-words has the formulation of:

$$\text{score}(\mathbf{t}|\mathbf{w}) = \lambda^\top f(\mathbf{t}, \mathbf{w})$$

- This can be decomposed into sum of features:

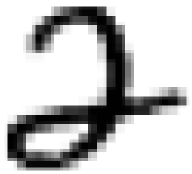
$$\lambda^\top \sum_i f(t_i, t_{i-1}, \mathbf{w}, i)$$

- Hence, we can use perceptron or MIRA style algorithms to train these models and learn the feature weights!

Feature Vectors

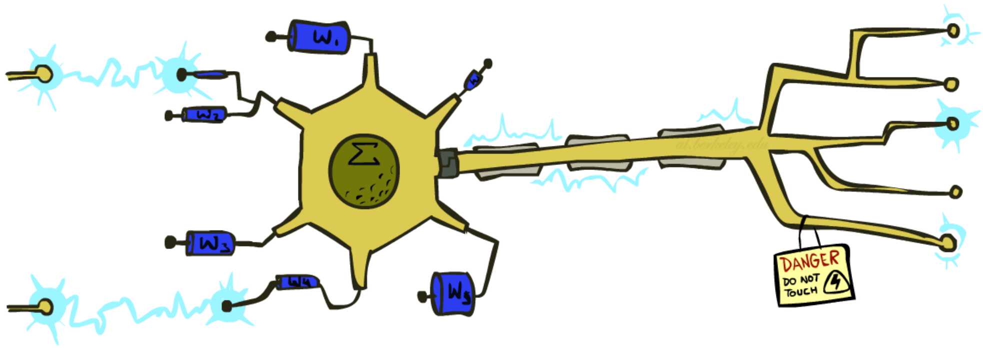
x	$f(x)$	y

Hello,
 	
Do you want free printr cartriges? Why pay more when you can get them ABSOLUTELY FREE! Just
	`# free : 2`
	`YOUR_NAME : 0`
	`MISSPELLED : 2`
	`FROM_FRIEND : 0`
	`...`
		SPAM
		or
		+

	`PIXEL-7,12 : 1`
	`PIXEL-7,13 : 0`
	`...`
	`NUM_LOOPS : 1`
	`...`
		“2”

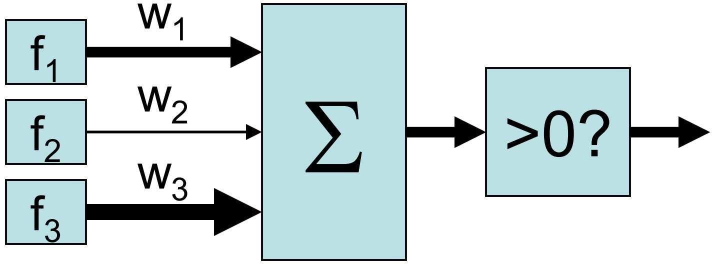
Linear Classifiers

- Inputs are feature values
- Each feature has a weight
- Sum is the activation



$$\text{activation}_w(x) = \sum_i w_i \cdot f_i(x) = w \cdot f(x)$$

- If the activation is:
  - Positive, output +1
  - Negative, output -1

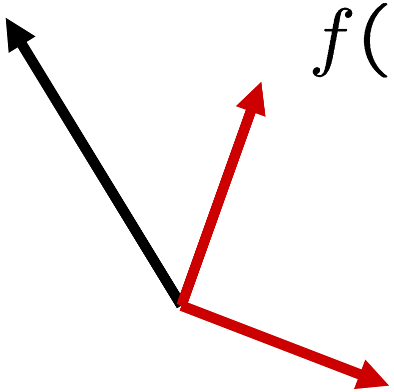


Weights

- Binary case: compare features to a weight vector
- Learning: figure out the weight vector from examples

```
# free      : 4
YOUR_NAME   :-1
MISSPELLED  : 1
FROM_FRIEND :-3
...
```

w



$f(x_1)$

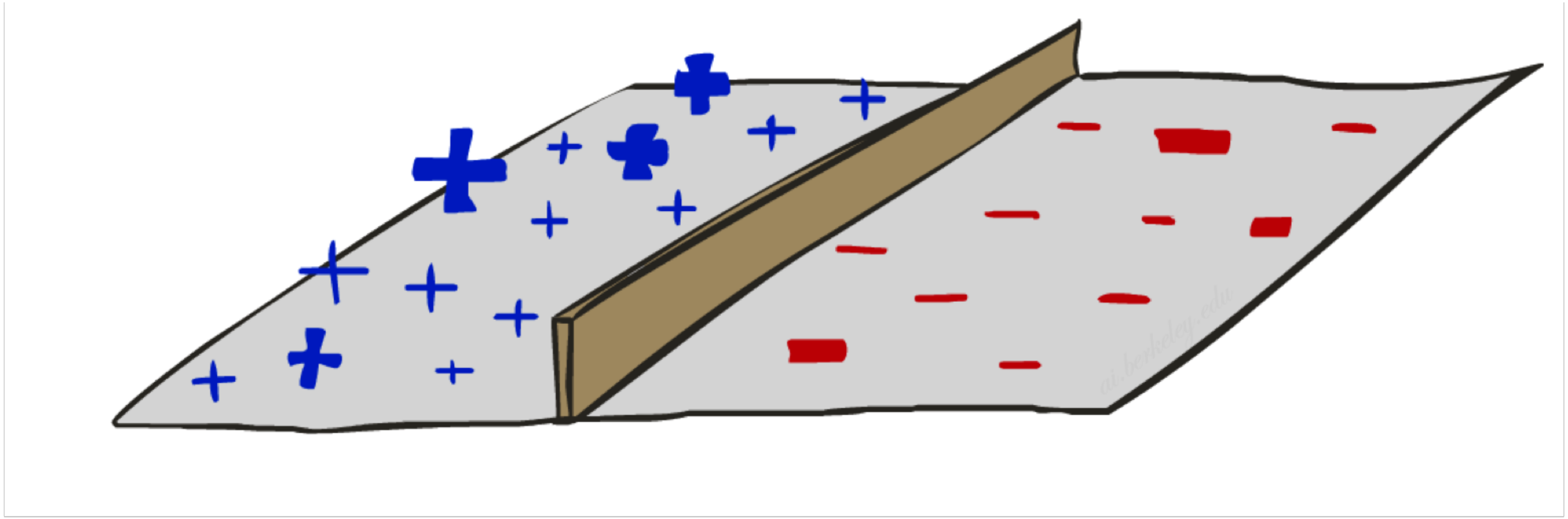
```
# free      : 2
YOUR_NAME   : 0
MISSPELLED  : 2
FROM_FRIEND : 0
...
```

$f(x_2)$

```
# free      : 0
YOUR_NAME   : 1
MISSPELLED  : 1
FROM_FRIEND : 1
...
```

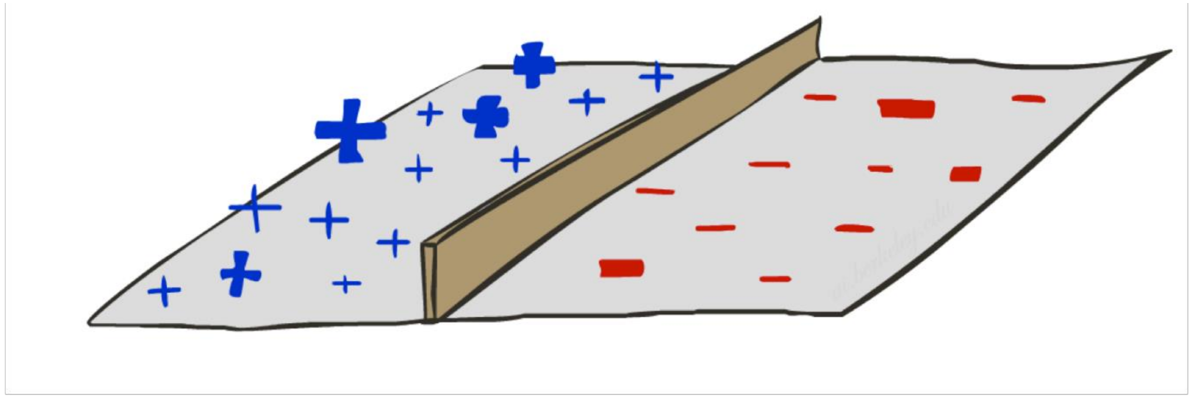
Dot product $w \cdot f$ *positive means the positive class*

Decision Rules



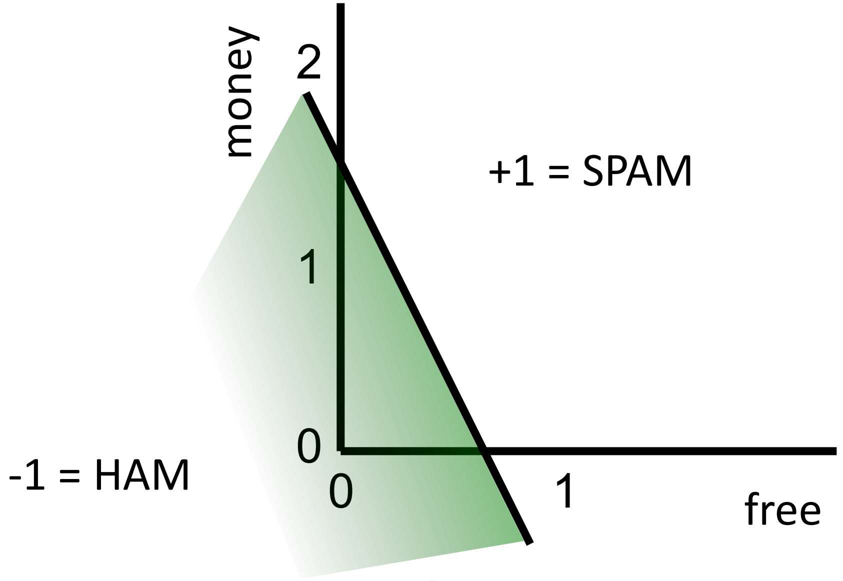
Binary Decision Rule

- In the space of feature vectors
  - Examples are points
  - Any weight vector is a hyperplane
  - One side corresponds to Y=+1
  - Other corresponds to Y=-1



$$w$$

```
BIAS  : -3
free  :  4
money :  2
...
```

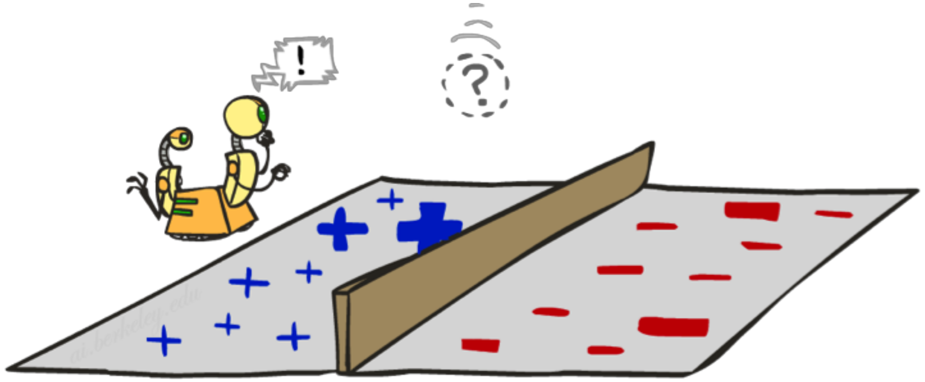


$$f \cdot w = 0$$

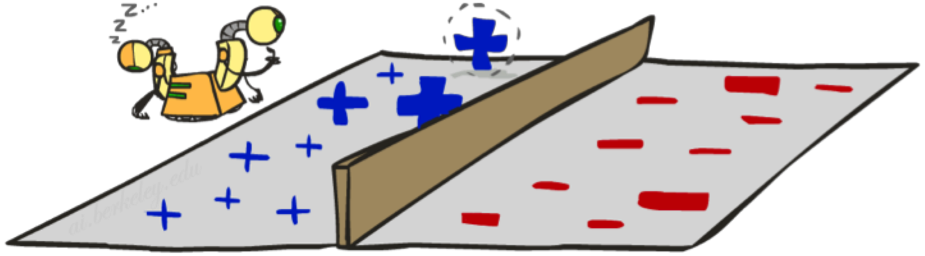
Weight Updates

Learning: Binary Perceptron

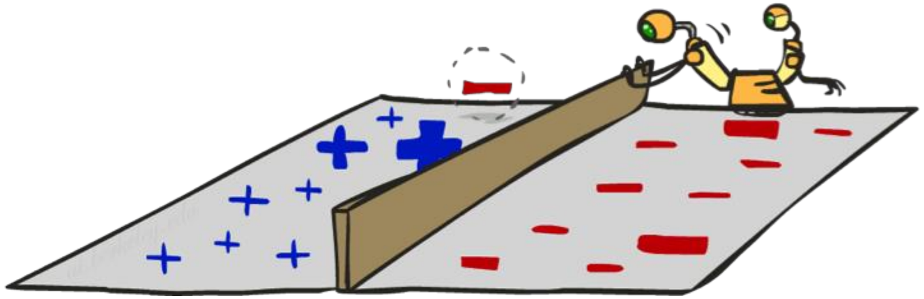
- Start with weights = 0
- For each training instance:
  - Classify with current weights



  - If correct (i.e., y=y*), no change!



  - If wrong: adjust the weight vector



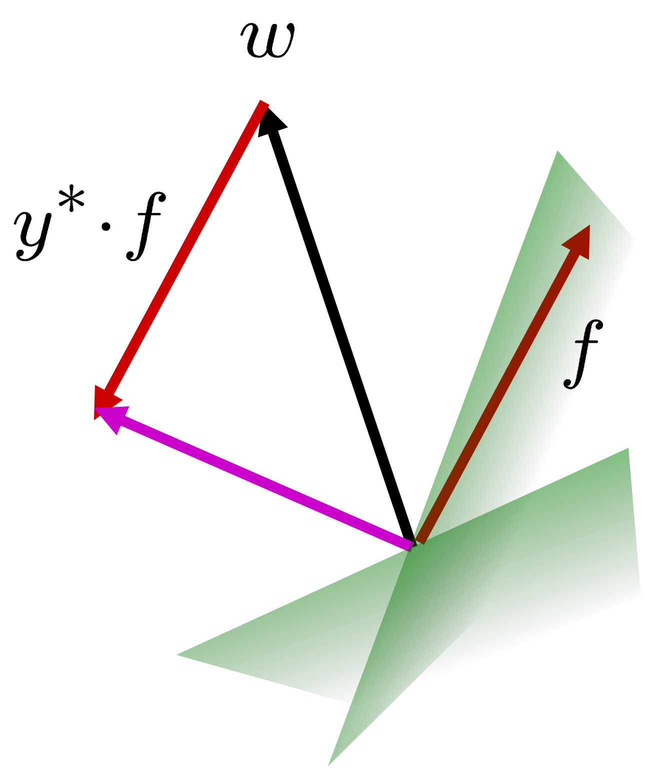
Learning: Binary Perceptron

- Start with weights = 0
- For each training instance:
  - Classify with current weights

$$y = \begin{cases} +1 & \text{if } \; w \cdot f(x) \geq 0 \\ -1 & \text{if } \; w \cdot f(x) < 0 \end{cases}$$

  - If correct (i.e., y=y*), no change!
  - If wrong: adjust the weight vector by adding or subtracting the feature vector. Subtract if y* is -1.

$$w = w + y^* \cdot f$$



Perceptron Training Algorithm

[Collins 2001]

Inputs: Training examples (x_i, y_i)
Initialization: Set $\bar{\alpha} = 0$
Algorithm:
For $t = 1 \ldots T$, $i = 1 \ldots n$
Calculate $z_i = \arg\max_{z \in \mathbf{GEN}(x_i)} \Phi(x_i, z) \cdot \bar{\alpha}$
If$(z_i \neq y_i)$ then $\bar{\alpha} = \bar{\alpha} + \Phi(x_i, y_i) - \Phi(x_i, z_i)$
Output: Parameters $\bar{\alpha}$

Conditional Random Field (CRF) Tagger

- MEMM

$$P(\mathbf{t}|\mathbf{w}) = \prod_i \frac{1}{Z(i)} \exp\left(\lambda^\top f(t_i, t_{i-1}, \mathbf{w}, i)\right)$$

- CRF

$$\begin{aligned} P(\mathbf{t}|\mathbf{w}) &= \frac{1}{Z(\mathbf{w})} \exp\left(\lambda^\top f(\mathbf{t}, \mathbf{w})\right) \\ &= \frac{1}{Z(\mathbf{w})} \exp\left(\lambda^\top \sum_i f(t_i, t_{i-1}, \mathbf{w}, i)\right) \\ &= \frac{1}{Z(\mathbf{w})} \prod_i \phi_i(t_i, t_{i-1}) \end{aligned}$$

CRF Training

- Derivatives needed have the form of "feature counts minus expected feature counts":

$$\frac{\partial L(\lambda)}{\partial \lambda} = \sum_k \left(\mathbf{f}_k(\mathbf{t}^k) - \sum_{\mathbf{t}} P(\mathbf{t}|\mathbf{w}_k)\mathbf{f}_k(\mathbf{t}) \right)$$

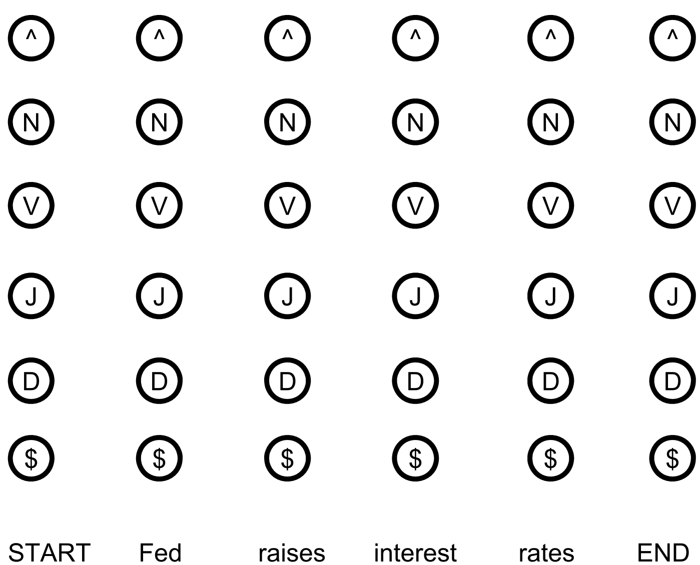
- These expected feature counts (under model distribution) in turn need posterior marginals:

$$\mathsf{count}(w, s) = \sum_{i:w_i=w} P(t_i = s|\mathbf{w})$$

$$\mathsf{count}(s \rightarrow s') = \sum_i P(t_{i-1} = s, t_i = s'|\mathbf{w})$$

Posterior Marginals

- And these posterior marginals in turn need the state trellis traversal similar to forward-backward discussed for HMM training:
  - How to compute that marginal?



$$\alpha_i(s) = \sum_{s'} \phi_i(s', s)\alpha_{i-1}(s')$$

$$\beta_i(s) = \sum_{s'} \phi_{i+1}(s, s')\beta_{i+1}(s')$$

$$P(t_i = s|\mathbf{w}) = \frac{\alpha_i(s)\beta_i(s)}{\alpha_N(\text{END})}$$

POS Tagging: Other Models

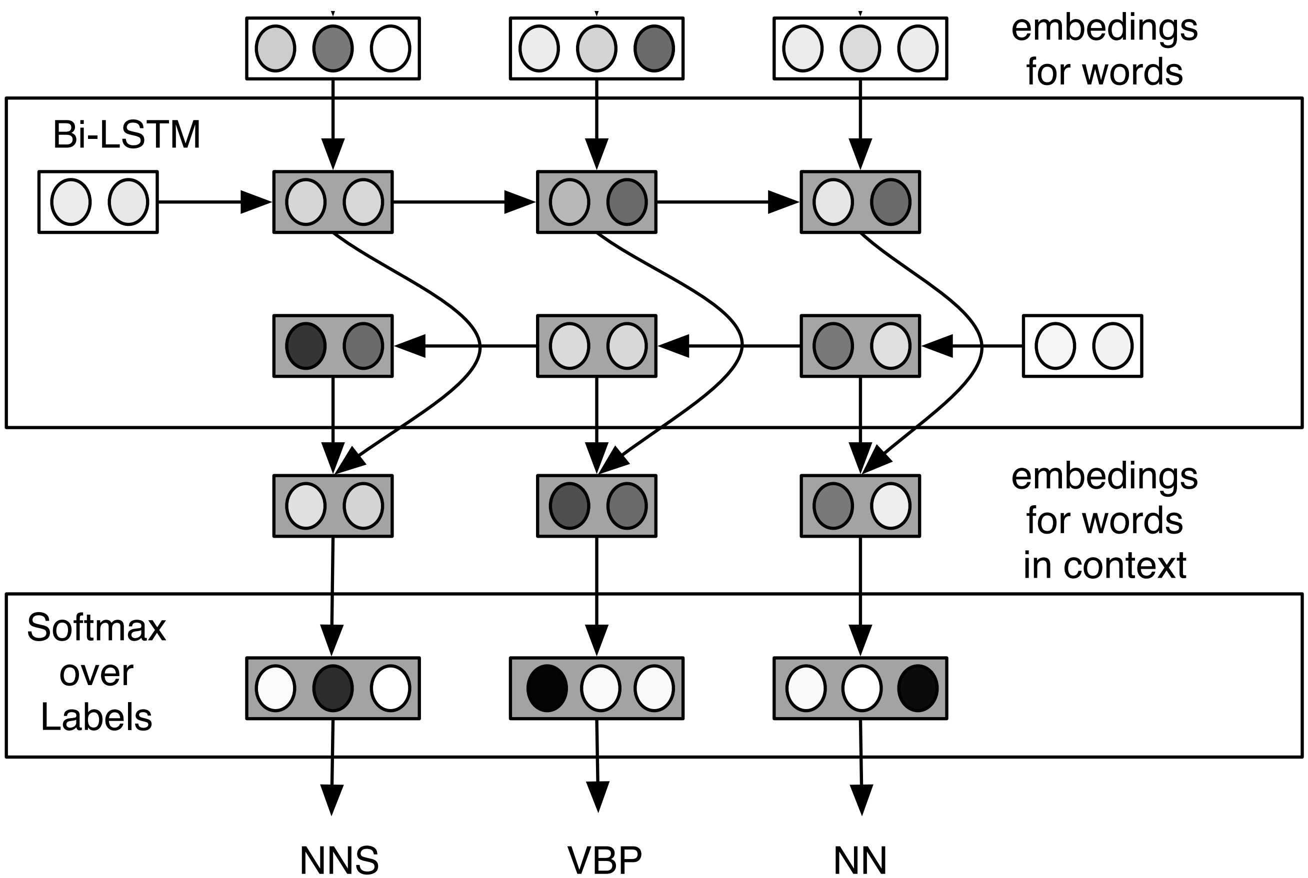
- Universal POS tagset for multilingual and cross-lingual tagging and parsing [Petrov et al., 2012]

  12 tags: NOUN, VERB, ADJ, ADV, PRON, DET, ADP, NUM, CONJ, PRT, ., X

- Unsupervised tagging also works reasonably well!
  [Yarowsky et al., 2001; Xi and Hwa, 2005; Berg-Kirkpatrick et al., 2010; Christodoulopoulos et al., 2010; Das and Petrov, 2011]

RNN-based POS-Tagger

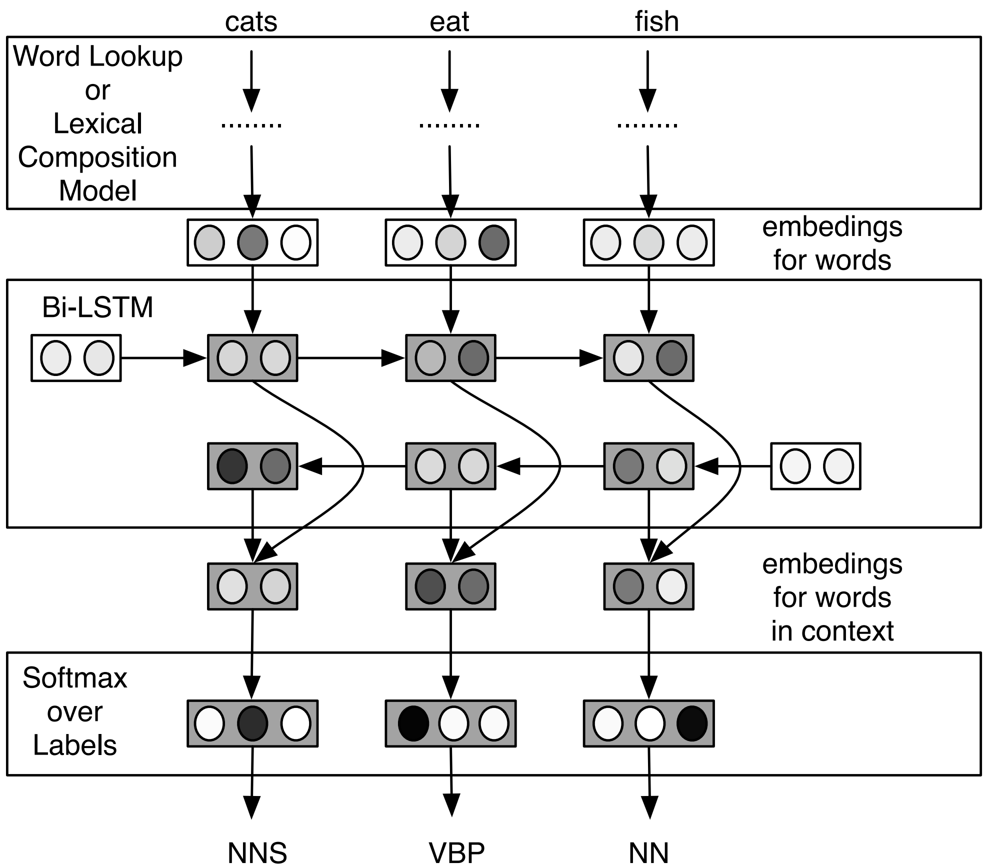
- Context captured by bidirectional LSTM; softmax on tag labels



[Ling et al., 2015 (and others)]

Char-RNN-based POS-Tagger

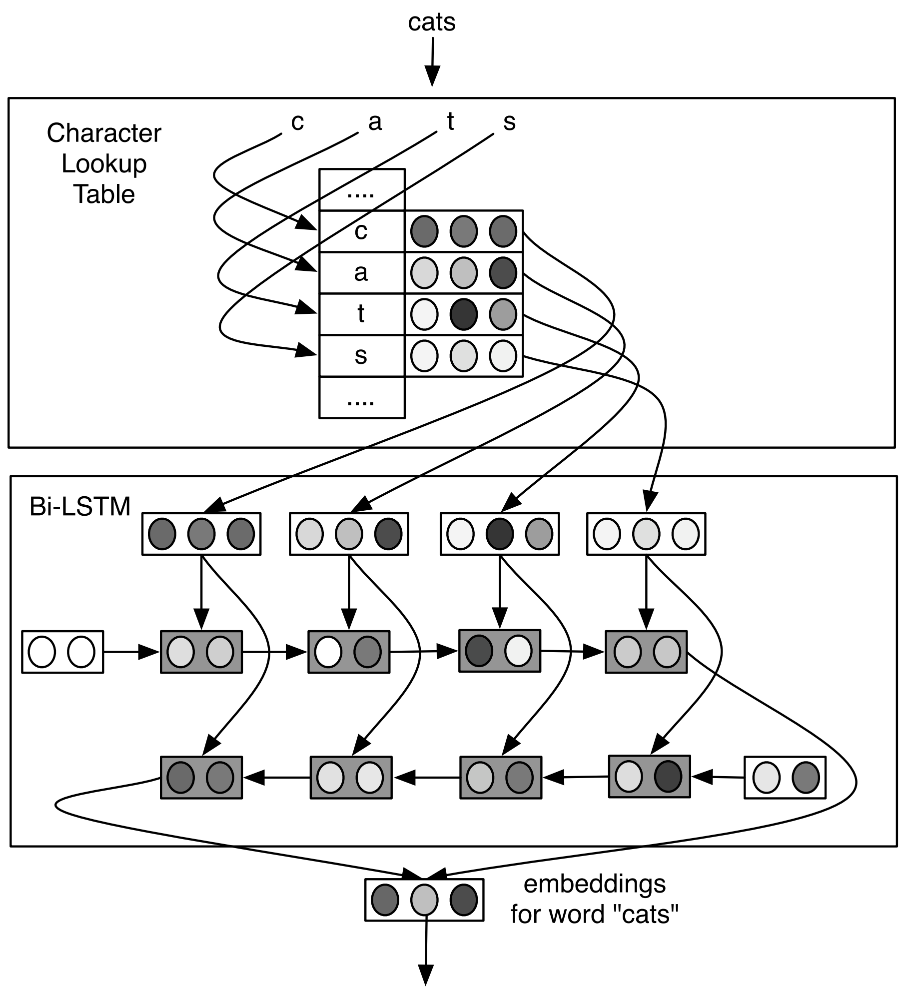
- Use character-based RNNs to compose word embeddings (to learn function)



[Ling et al., 2015 (and others)]

Char-RNN-based POS-Tagger

- Use character-based RNNs to compose word embeddings (to learn function)



[Ling et al., 2015 (and others)]

Other Sequence Labeling Tasks

- Named Entity Recognition
- Spelling Correction
- Word Alignment
- Noun Phrase Chunking
- Supersense Tagging
- Multiword Expressions

Named Entity Recognition

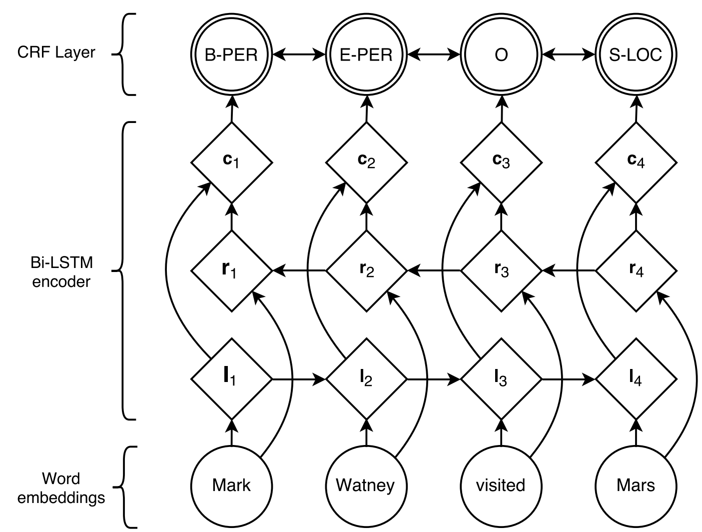
- Label proper nouns as person, location, organization, other

PER PER O O O O O O ORG O O O O O LOC LOC O

Tim Boon has signed a contract extension with Leicestershire which will keep him at Grace Road .

- Also prefers rich contextual features
- CRF models perform strongly for this
- Neural+CRF versions even stronger →

[Lample et al., 2016]



[Bikel et al., 1999]

Fine-Grained NER

PERSON

- **artist**
  - actor
  - author
  - director
  - music
- **education**
  - student
  - teacher
- **athlete**
- **business**
- **coach**
- **doctor**
- **legal**
- **military**
- **political figure**
- **religious leader**
- **title**

LOCATION

- **structure**
  - airport
  - government
  - hospital
  - hotel
  - restaurant
  - sports facility
  - theatre
- **geography**
  - body of water
  - island
  - mountain
- **transit**
  - bridge
  - railway
  - road
- **celestial**
- **city**
- **country**
- **park**

ORGANIZATION

- **company**
  - broadcast
  - news
- **education**
- **government**
- **military**
- **music**
- **political party**
- **sports league**
- **sports team**
- **stock exchange**
- **transit**

OTHER

- **art**
  - broadcast
  - film
  - music
  - stage
  - writing
- **event**
  - accident
  - election
  - holiday
  - natural disaster
  - protest
  - sports event
  - violent conflict
- **health**
  - malady
  - treatment
- **award**
- **body part**
- **currency**
- **language**
  - programming language
- **living thing**
  - animal
- **product**
  - camera
  - car
  - computer
  - mobile phone
  - software
  - weapon
- **food**
- **heritage**
- **internet**
- **legal**
- **religion**
- **scientific**
- **sports & leisure**
- **supernatural**

[Gillick et al., 2014]

Fine-Grained NER

person	doctor
actor	engineer
architect	monarch
artist	musician
athlete	politician
author	religious_leader
coach	soldier
director	terrorist

organization	terrorist_organization
airline	government_agency
company	government
educational_institution	political_party
fraternity_sorority	educational_department
sports_league	military
sports_team	news_agency

location	body_of_water
city	island
country	mountain
county	glacier
province	astral_body
railway	cemetery
road	park
bridge	

product	camera
engine	mobile_phone
airplane	computer
car	software
ship	game
spacecraft	instrument
train	weapon

art	written_work
film	newspaper
play	music

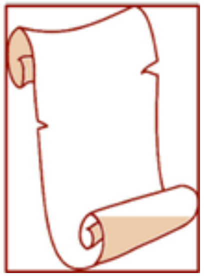
event	military_conflict
attack	natural_disaster
election	sports_event
protest	terrorist_attack

building	time	chemical_thing	website
airport	color	biological_thing	broadcast_network
dam	award	medical_treatment	broadcast_program
hospital	educational_degree	disease	tv_channel
hotel	title	symptom	currency
library	law	drug	stock_exchange
power_station	ethnicity	body_part	algorithm
restaurant	language	living_thing	programming_language
sports_facility	religion	animal	transit_system
theater	god	food	transit_line

[Ling and Weld, 2012]

Coreference Resolution

Coreference Resolution

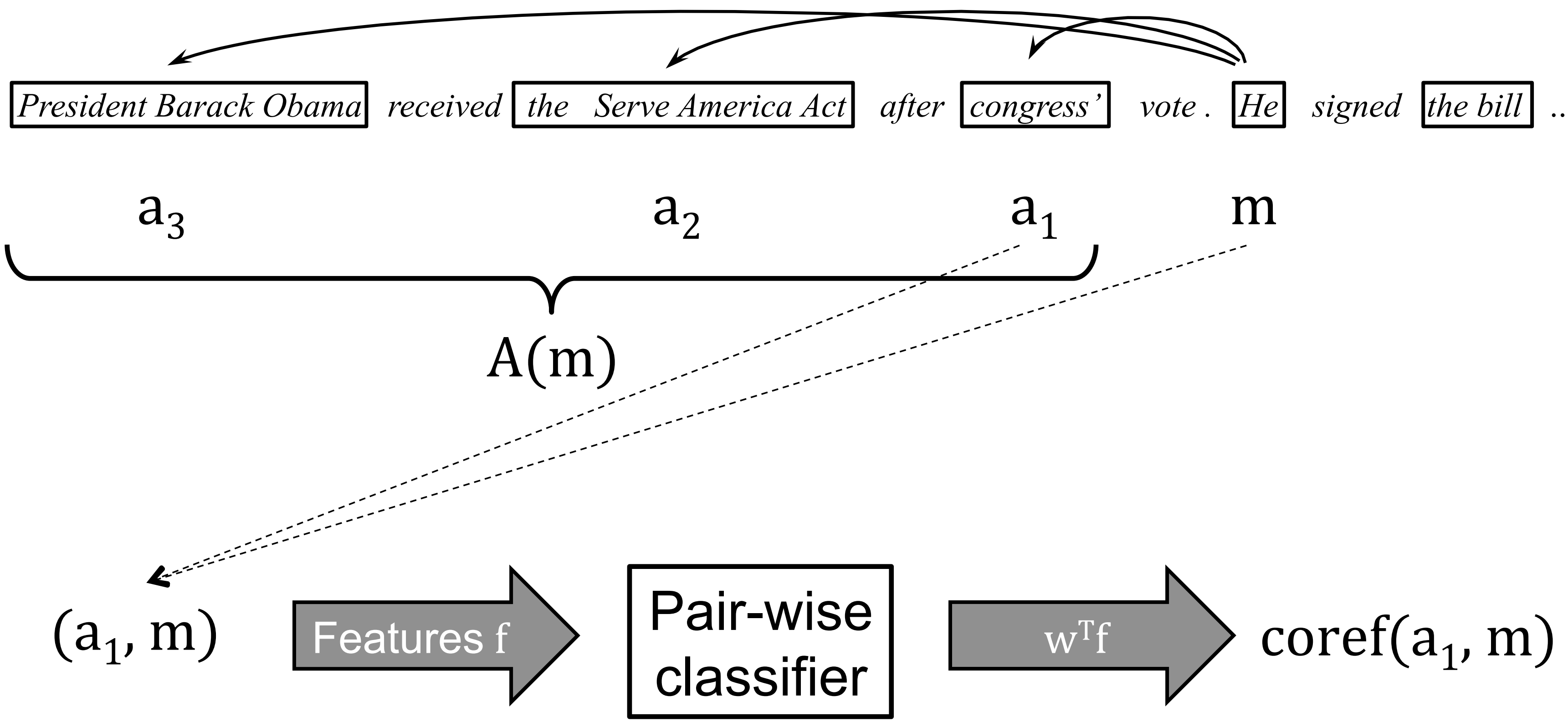
   

President Barack Obama received the Serve America Act after congress' vote. He signed the bill last Thursday. The president said it would greatly increase service opportunities for the American people.

- Mentions to entity/event clusters
- Demos: http://nlp.stanford.edu:8080/corenlp/process

Mention-pair Models

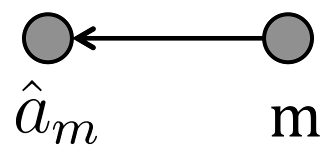
- Pair-wise classification approach:

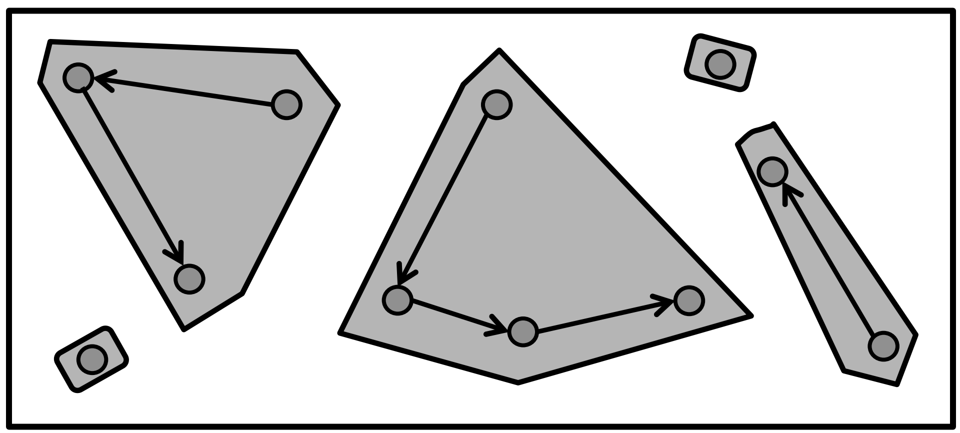


[Soon et al. 2001, Ng and Cardie 2002; Bengtson and Roth, 2008; Stoyanov et al., 2010]

Mention-pair Model

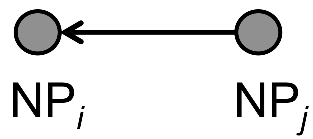
For each mention m, $\hat{a}_m = \underset{a_i \in A(m)}{\operatorname{argmax}}\; coref(a_i, m)$





[Soon et al. 2001, Ng and Cardie 2002; Bengtson and Roth, 2008; Stoyanov et al., 2010]

Standard features



Type	Feature	Description
LEXICAL	SOON_STR	Do the strings match after removing determiners ?
GRAMMATICAL	NUMBER	Do NP_i and NP_j agree in number ?
	GENDER	Do NP_i and NP_j agree in gender ?
	APPOSITIVE	Are the NPs in an appositive relationship ?
SEMANTIC	WORDNET_CLASS	Do NP_i and NP_j have the same WordNet class ?
	ALIAS	Is one NP an alias of the other ?
POSITIONAL	SENTNUM	Distance between the NPs in terms of # of sentences

- Weaknesses: All pairs, Transitivity/Independence errors (*He – Obama – She*), Insufficient information

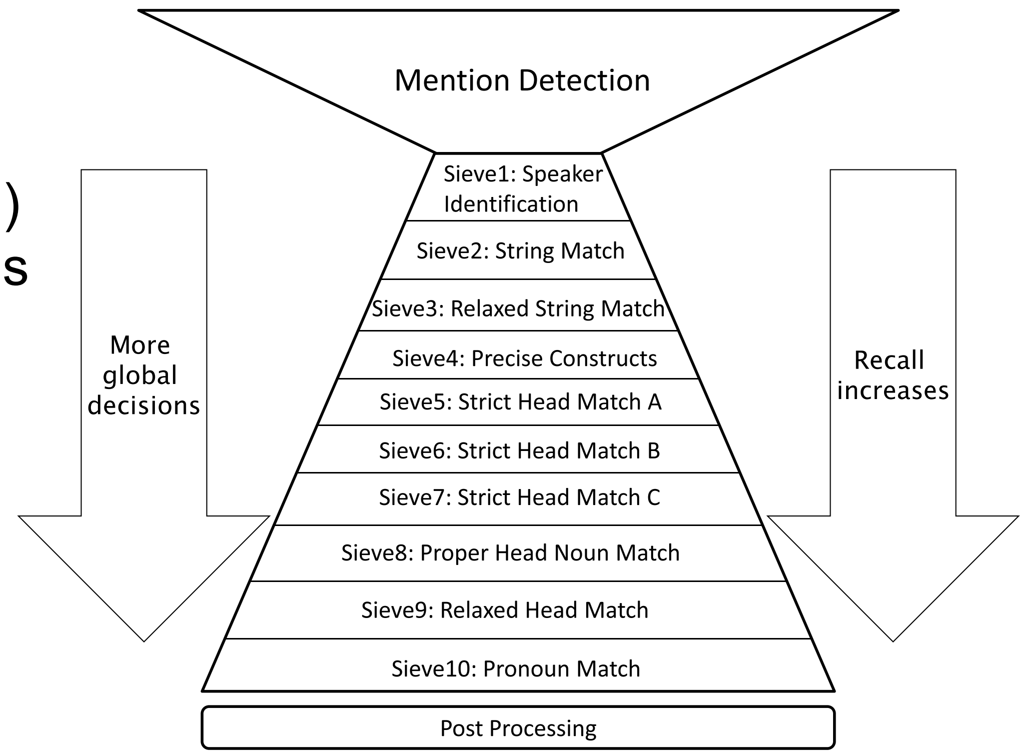
[Soon et al. 2001, Ng and Cardie 2002; Bengtson and Roth, 2008; Stoyanov et al., 2010]

Entity-centric Models

- Each coreference decision is globally informed by previously clustered mentions and their shared attributes

- Lee et al., 2013's deterministic (rule-based) system: multiple, cautious sieves from high to low precision

- Durrett et al., 2013's entity-level model is discriminative, probabilistic using factor graphs and BP

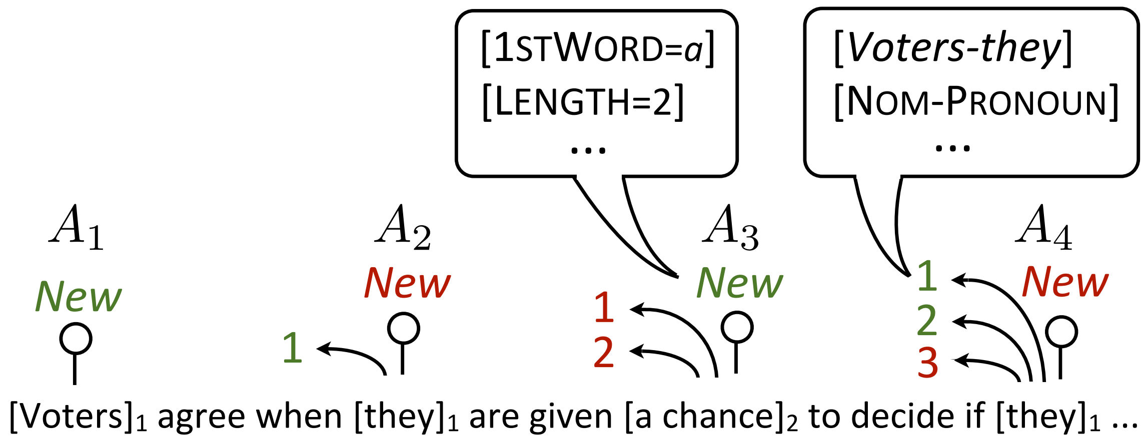


[Haghighi and Klein, 2009; Lee et al., 2013; Durrett et al., 2013]

Mention-Ranking Models (Learned)

- Log-linear model to select at most 1 antecedent for each mention or determine that it begins a new cluster

$$Pr(A_i = a|x) \propto \exp(w^\top f(i, a, x))$$



[Denis and Baldridge, 2008; Durrett and Klein, 2013]

- Recent work (Wiseman et al., 2016, Clark & Manning, 2016) has used NNs for non-linear and vector-space coreference features to achieve SoA!

Adding Knowledge to Coref

- External corpora: Web, Wikipedia, YAGO, FrameNet, Gender/Number/Person lists/classifiers, 3D Images, Videos

- Methods:
  - Self-training, Bootstrapping
  - Co-occurrence, Distributional, and Pattern-based Features
  - Entity Linking
  - Visual Cues from 3D Images and Videos

- Daumé III and Marcu, 2005; Markert and Nissim, 2005; Bergsma and Lin, 2006; Ponzetto and Strube, 2006; Haghighi and Klein, 2009; Kobdani et al., 2011; Rahman and Ng, 2011; Bansal and Klein, 2012; Durrett and Klein, 2014; Kong et al., 2014; Ramanathan et al., 2014

Web Features for Coreference

count(*Obama * president*) vs count(*Jobs * president*)

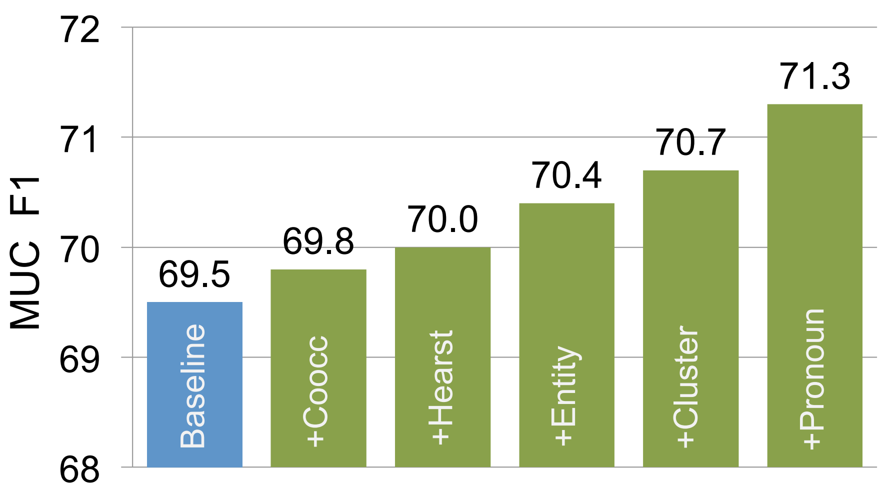
When Obama met Jobs , the president discussed the …

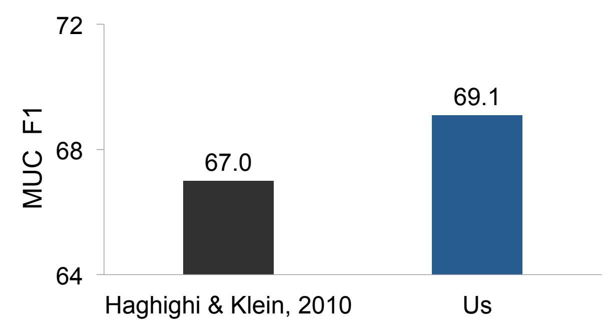
[Bansal and Klein, 2012]

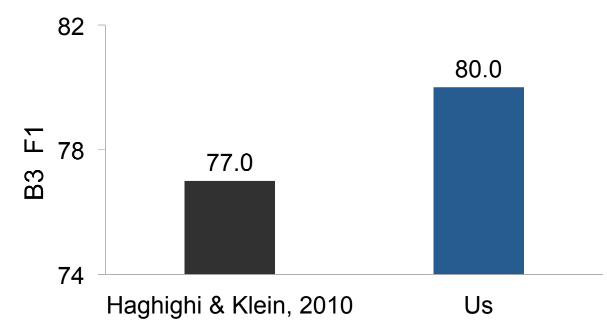
Web Features for Coreference

count(*Obama signed bills*) vs count(*Jobs signed bills*)

When Obama met Jobs , the … He signed bills that …



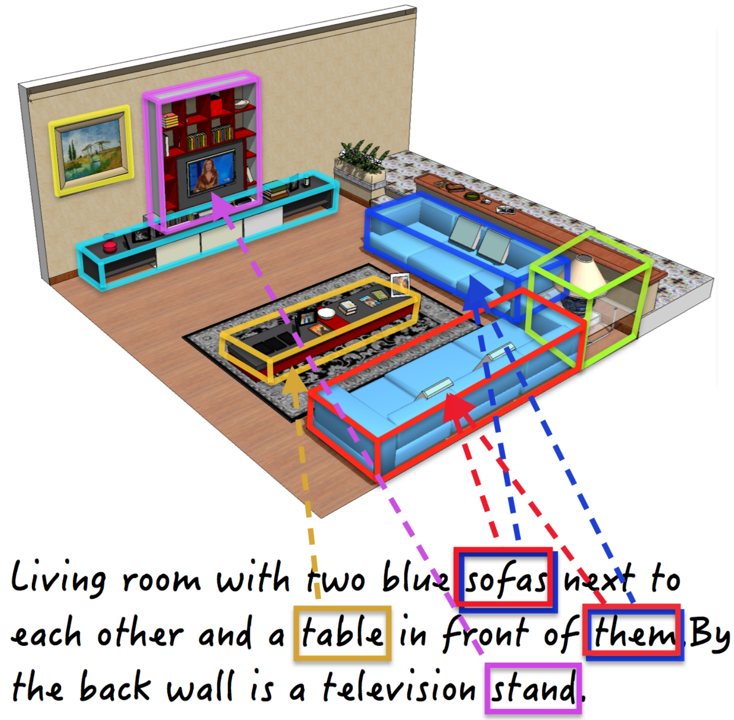




[Bansal and Klein, 2012]

Visual Cues for Coreference

- Joint coreference and 3D image recognition

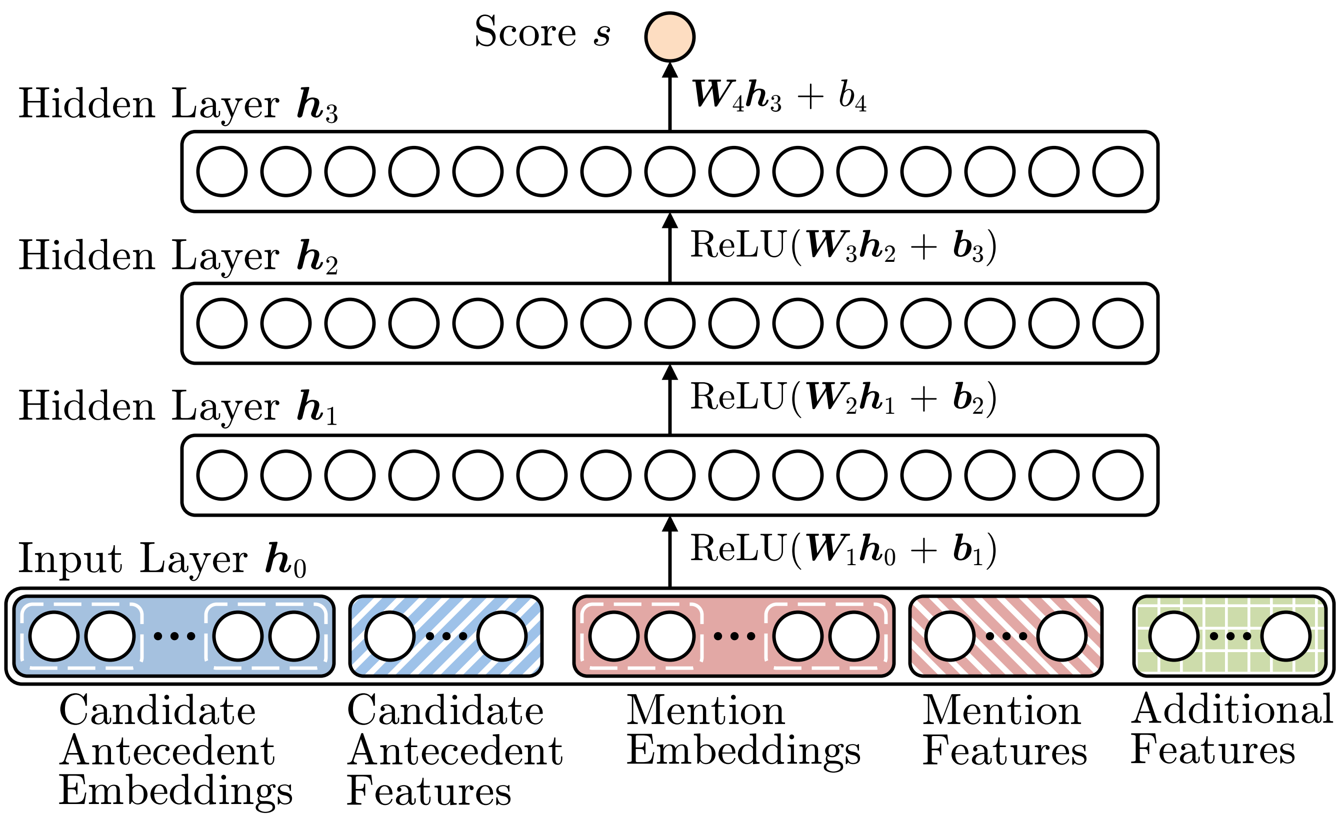


	MUC			B^3		
Method	precision	recall	F1	precision	recall	F1
Stanford	61.56	62.59	62.07	75.05	76.15	75.59
Ours	83.69	51.08	63.44	88.42	70.02	78.15

[Kong, Lin, Bansal, Urtasun, and Fidler, 2014]

Neural Models for Coreference

- Mention-pair model as simple feed-forward network:



[Clark and Manning, 2016; Wiseman et al., 2015]