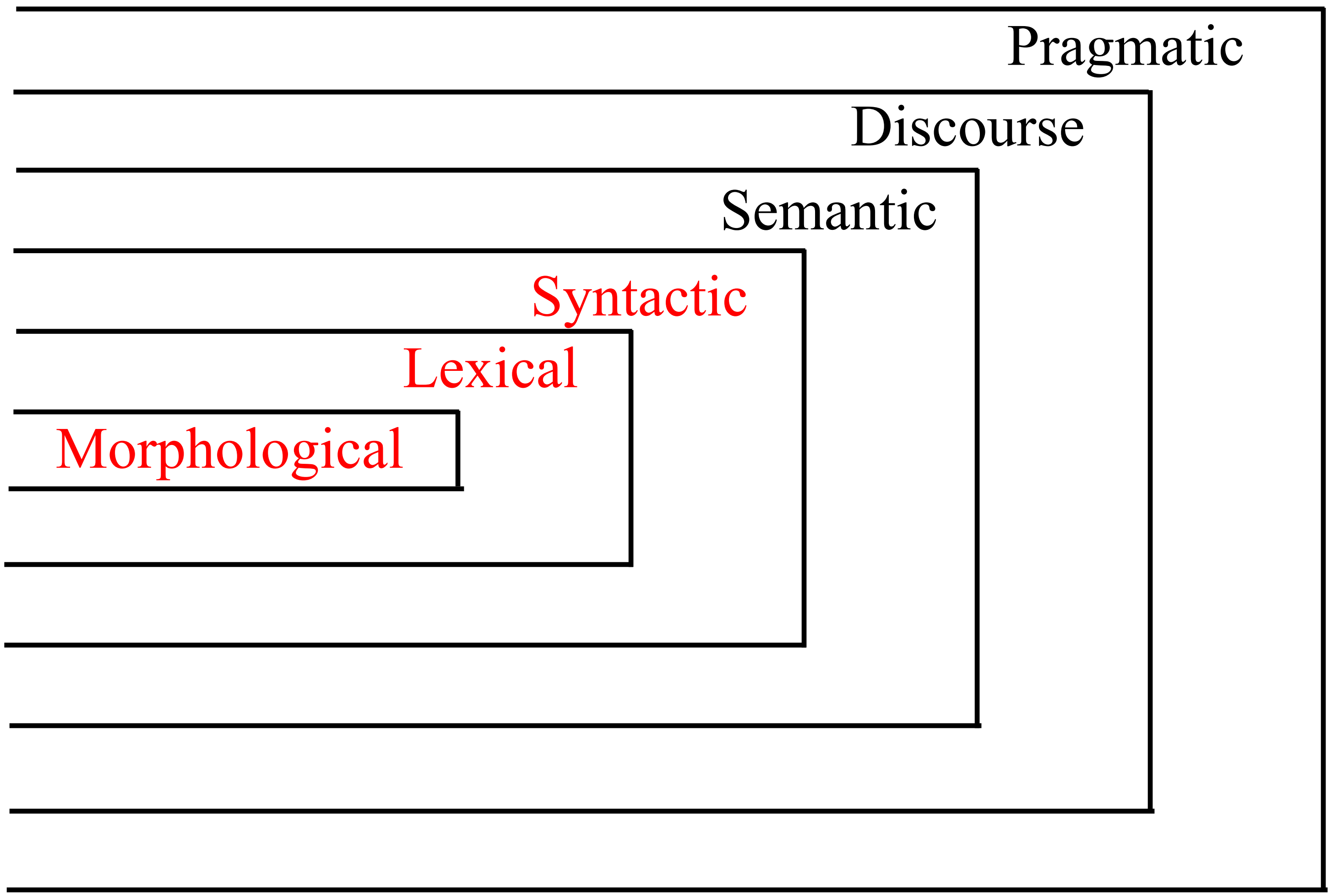
Part-Of-Speech (POS) Tagging

Synchronic Model of Language



What is Part-Of-Speech Tagging?

- The general purpose of a part-of-speech tagger is to associate each word in a text with its correct lexical-syntactic category (represented by a tag)

  03/14/1999 (AFP)… the extremist Harkatul Jihad group, reportedly backed by Saudi dissident Osama bin Laden ...

  … the|DT extremist|JJ Harkatul|NNP Jihad|NNP group|NN ,|, reportedly|RB backed|VBD by|IN Saudi|NNP dissident|NN Osama|NNP bin|NN Laden|NNP …

What are Parts-of-Speech?

- Approximately 8 traditional basic word classes, sometimes called lexical classes or types
- These are the ones taught in grade school grammar

N	noun	*chair, bandwidth, pacing*
V	verb	*study, debate, munch*
ADJ	adjective	*purple, tall, ridiculous (includes articles)*
ADV	adverb	*unfortunately, slowly*
P	preposition	*of, by, to*
CON	conjunction	*and, but*
PRO	pronoun	*I, me, mine*
INT	interjection	*um*

Classes for Open Class Words

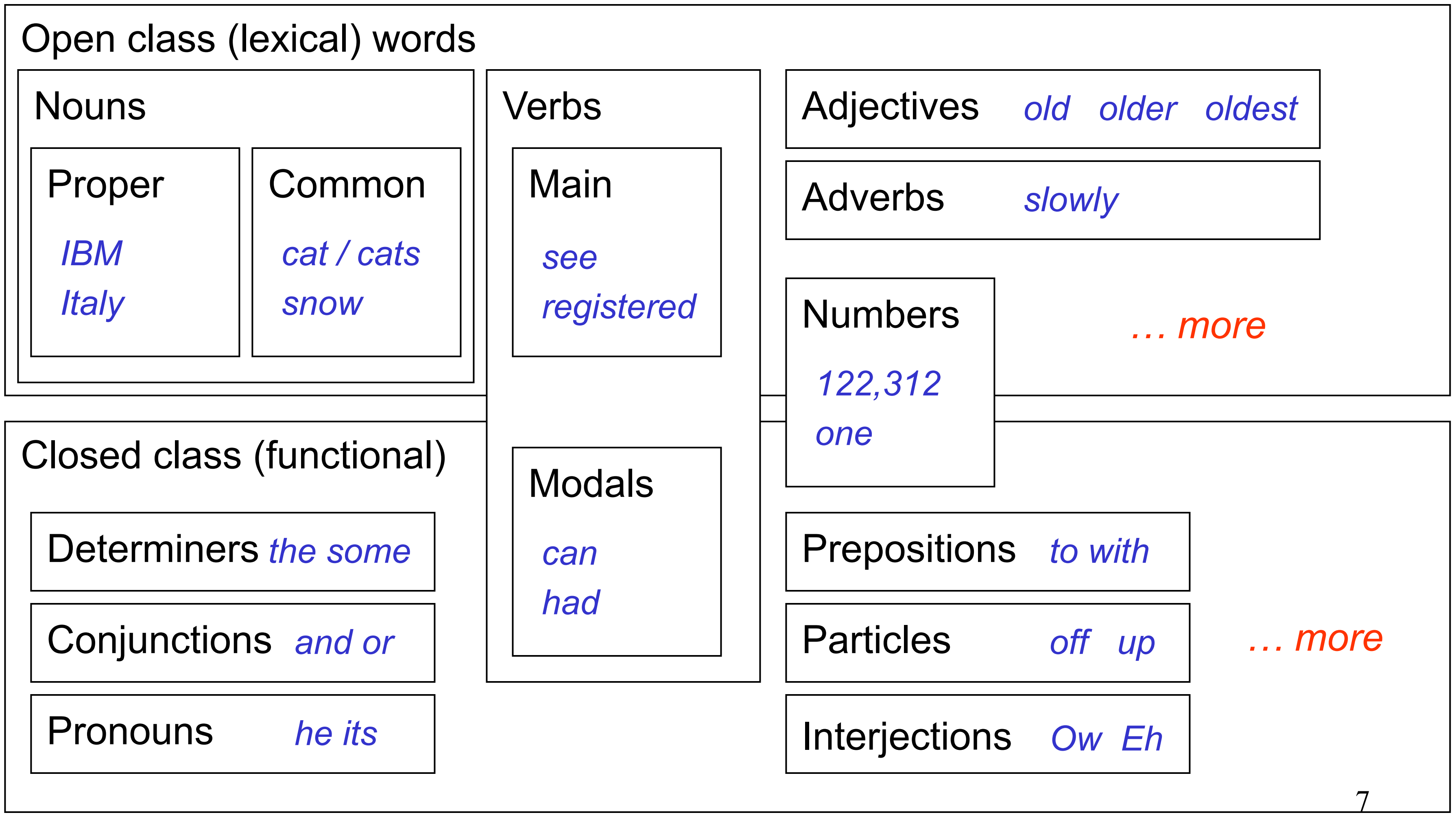
- Open classes – can add words to these basic word classes:
    - Nouns, Verbs, Adjectives, Adverbs.
        - Every known human language has nouns and verbs
- Nouns: people, places, things
    - Classes of nouns
        - proper vs. common
        - count vs. mass
    - Properties of nouns: can be preceded by a determiner, etc.
- Verbs: actions and processes
- Adjectives: properties, qualities
- Adverbs: hodgepodge!
    - Unfortunately, John walked home extremely slowly yesterday
- Numerals: one, two, three, third, …

Classes for Closed Class Words

- Closed classes– words are not added to these classes:
  - determiners: a, an, the
  - pronouns: she, he, I
  - prepositions: on, under, over, near, by, …
    - over the river and through the woods
  - particles: up, down, on, off, …
    - Used with verbs and have slightly different meaning than when used as a preposition
      - she turned the paper over
- Closed class words are often function words which have structuring uses in grammar:
  - of, it , and , you
- Differ more from language to language than open class words

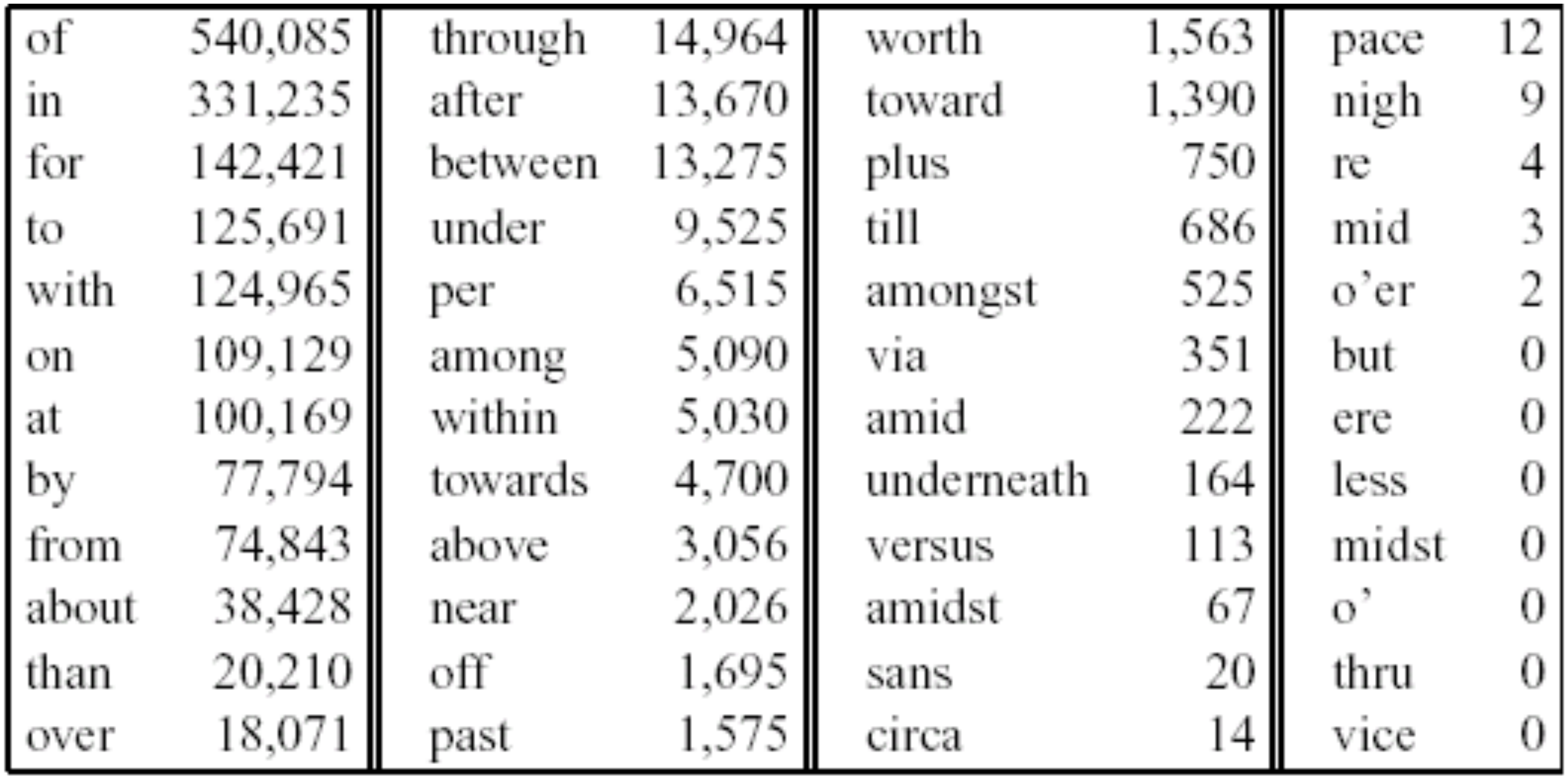
Open and Closed Classes

- We may want to make more distinctions than 8 classes:



Prepositions from CELEX

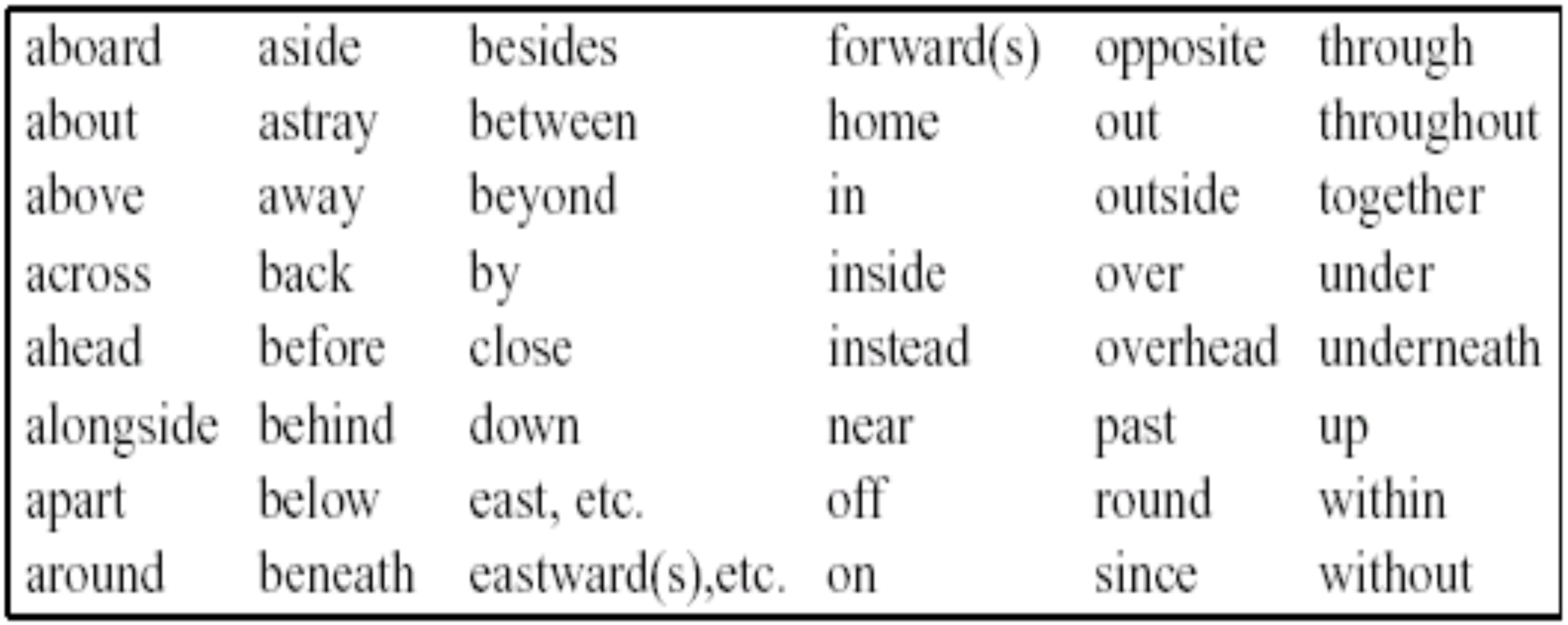
- From the CELEX on-line dictionary with frequencies from the COBUILD corpus



of	540,085	through	14,964	worth	1,563	pace	12
in	331,235	after	13,670	toward	1,390	nigh	9
for	142,421	between	13,275	plus	750	re	4
to	125,691	under	9,525	till	686	mid	3
with	124,965	per	6,515	amongst	525	o'er	2
on	109,129	among	5,090	via	351	but	0
at	100,169	within	5,030	amid	222	ere	0
by	77,794	towards	4,700	underneath	164	less	0
from	74,843	above	3,056	versus	113	midst	0
about	38,428	near	2,026	amidst	67	o'	0
than	20,210	off	1,695	sans	20	thru	0
over	18,071	past	1,575	circa	14	vice	0

Charts from Jurafsky and Martin text

English Single-Word Particles



aboard	aside	besides	forward(s)	opposite	through
about	astray	between	home	out	throughout
above	away	beyond	in	outside	together
across	back	by	inside	over	under
ahead	before	close	instead	overhead	underneath
alongside	behind	down	near	past	up
apart	below	east, etc.	off	round	within
around	beneath	eastward(s),etc.	on	since	without

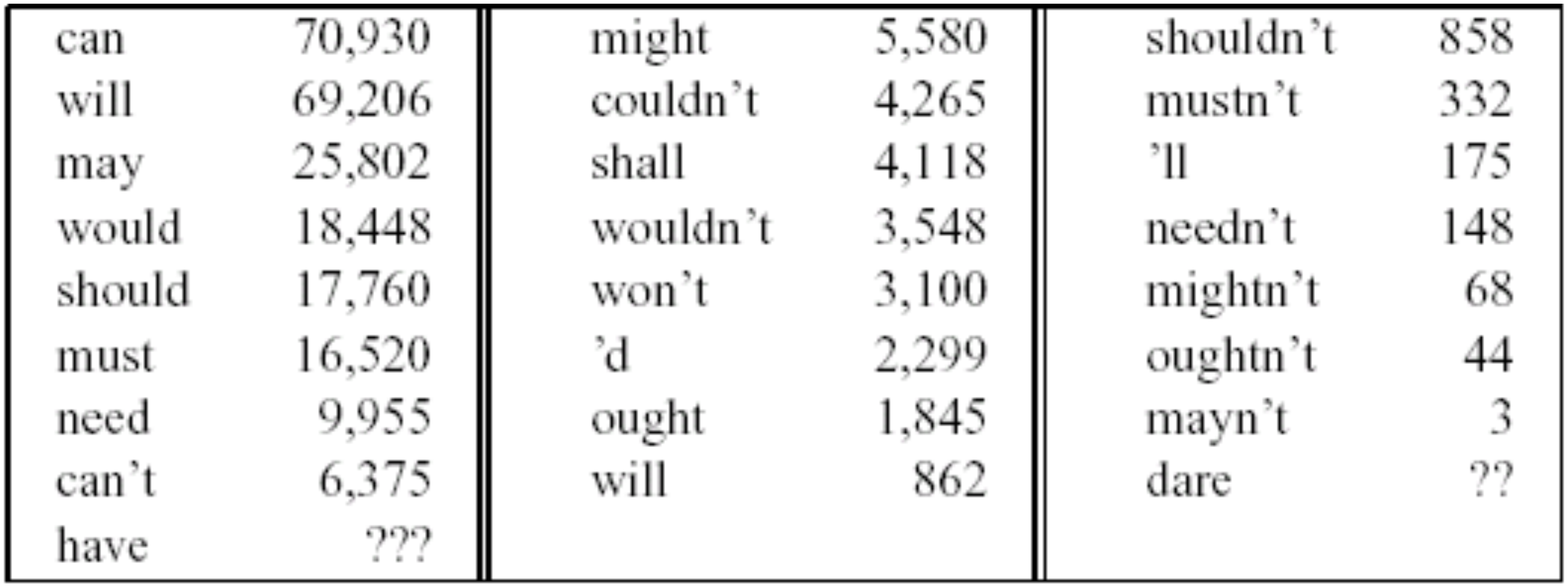
Pronouns in CELEX

it	199,920	how	13,137	yourself	2,437	no one	106
I	198,139	another	12,551	why	2,220	wherein	58
he	158,366	where	11,857	little	2,089	double	39
you	128,688	same	11,841	none	1,992	thine	30
his	99,820	something	11,754	nobody	1,684	summat	22
they	88,416	each	11,320	further	1,666	suchlike	18
this	84,927	both	10,930	everybody	1,474	fewest	15
that	82,603	last	10,816	ourselves	1,428	thyself	14
she	73,966	every	9,788	mine	1,426	whomever	11
her	69,004	himself	9,113	somebody	1,322	whosoever	10
we	64,846	nothing	9,026	former	1,177	whomsoever	8
all	61,767	when	8,336	past	984	wherefore	6
which	61,399	one	7,423	plenty	940	whereat	5
their	51,922	much	7,237	either	848	whatsoever	4
what	50,116	anything	6,937	yours	826	whereon	2
my	46,791	next	6,047	neither	618	whoso	2
him	45,024	themselves	5,990	fewer	536	aught	1
me	43,071	most	5,115	hers	482	howsoever	1
who	42,881	itself	5,032	ours	458	thrice	1
them	42,099	myself	4,819	whoever	391	wheresoever	1
no	33,458	everything	4,662	least	386	you-all	1
some	32,863	several	4,306	twice	382	additional	0
other	29,391	less	4,278	theirs	303	anybody	0
your	28,923	herself	4,016	wherever	289	each other	0
its	27,783	whose	4,005	oneself	239	once	0
our	23,029	someone	3,755	thou	229	one another	0
these	22,697	certain	3,345	'un	227	overmuch	0
any	22,666	anyone	3,318	ye	192	such and such	0
more	21,873	whom	3,229	thy	191	whate'er	0
many	17,343	enough	3,197	whereby	176	whenever	0
such	16,880	half	3,065	thee	166	whereof	0
those	15,819	few	2,933	yourselves	148	whereto	0
own	15,741	everyone	2,812	latter	142	whereunto	0
us	15,724	whatever	2,571	whichever	121	whichsoever	0

Conjunctions

and	514,946	yet	5,040	considering	174	forasmuch as	0
that	134,773	since	4,843	lest	131	however	0
but	96,889	where	3,952	albeit	104	immediately	0
or	76,563	nor	3,078	providing	96	in as far as	0
as	54,608	once	2,826	whereupon	85	in so far as	0
if	53,917	unless	2,205	seeing	63	inasmuch as	0
when	37,975	why	1,333	directly	26	insomuch as	0
because	23,626	now	1,290	ere	12	insomuch that	0
so	12,933	neither	1,120	notwithstanding	3	like	0
before	10,720	whenever	913	according as	0	neither nor	0
though	10,329	whereas	867	as if	0	now that	0
than	9,511	except	864	as long as	0	only	0
while	8,144	till	686	as though	0	provided that	0
after	7,042	provided	594	both and	0	providing that	0
whether	5,978	whilst	351	but that	0	seeing as	0
for	5,935	suppose	281	but then	0	seeing as how	0
although	5,424	cos	188	but then again	0	seeing that	0
until	5,072	supposing	185	either or	0	without	0

Auxiliary Verbs



can	70,930	might	5,580	shouldn’t	858
will	69,206	couldn’t	4,265	mustn’t	332
may	25,802	shall	4,118	’ll	175
would	18,448	wouldn’t	3,548	needn’t	148
should	17,760	won’t	3,100	mightn’t	68
must	16,520	’d	2,299	oughtn’t	44
need	9,955	ought	1,845	mayn’t	3
can’t	6,375	will	862	dare	??
have	???				

Possible Tag Sets for English

- Kucera & Brown (Brown Corpus) – 87 POS tags
- C5 (British National Corpus) – 61 POS tags
  - Tagged by Lancaster's UCREL project
- Penn Treebank – 45 POS tags
  - Most widely used of the tag sets today

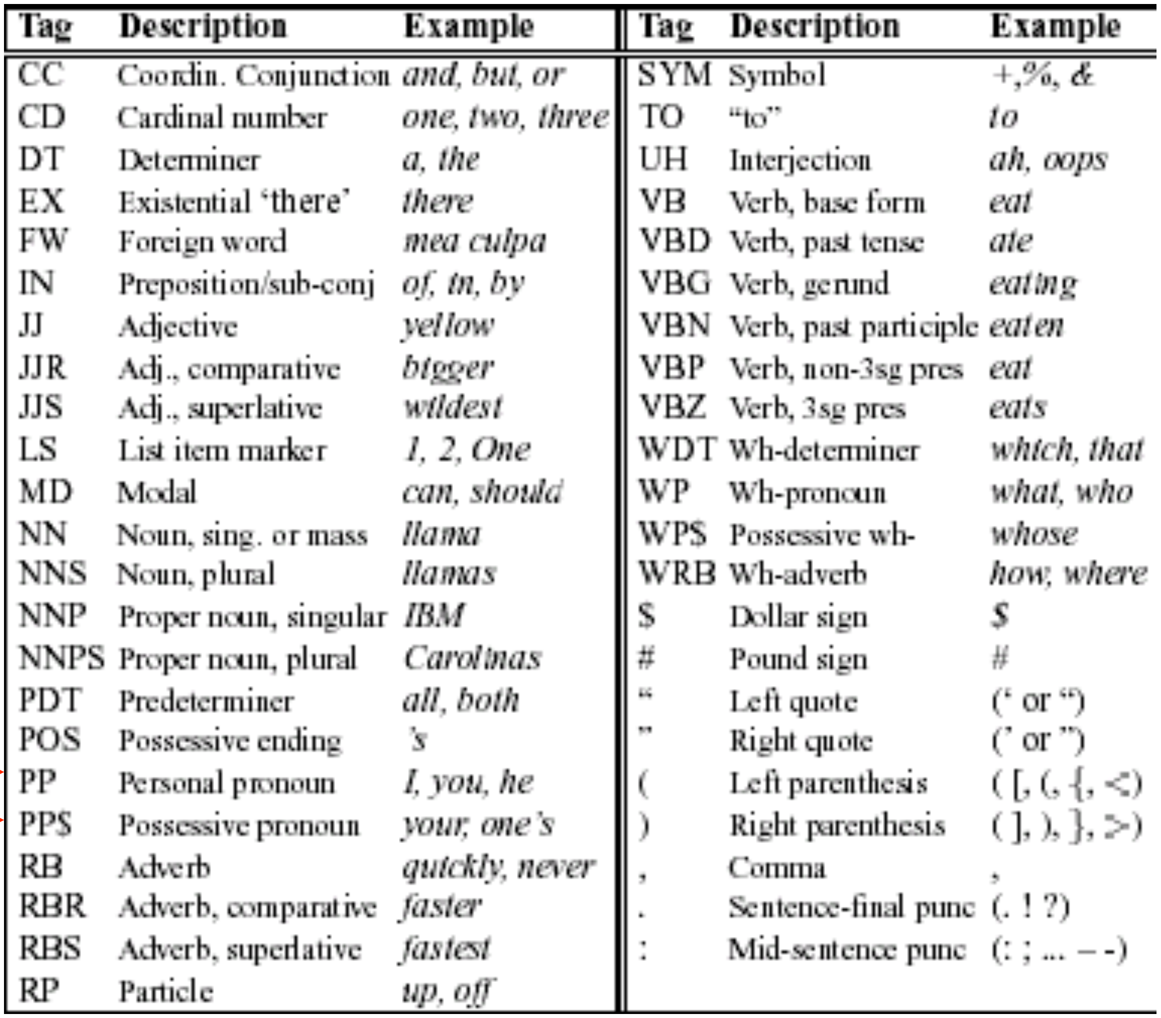
Penn Treebank

- A corpus containing:
  - over 1.6 million words of hand-parsed material from the Dow Jones News Service, plus an additional 1 million words tagged for part-of-speech.
  - the first fully parsed version of the Brown Corpus, which has also been completely retagged using the Penn Treebank tag set.
  - source code for several software packages which permits the user to search for specific constituents in tree structures.
- Costs $1,250 to $2,500 for research use
- Separate licensing needed for commercial use

Word Classes: Penn Treebank Tag Set

PRP

PRP$



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

Example of Penn Treebank Tagging of Brown Corpus Sentence

- The/DT grand/JJ jury/NN commented/VBD on/IN a/DT number/NN of/IN other/JJ topics/NNS ./.

- VB DT NN .
Book that flight .

- VBZ DT NN VB NN ?
Does that flight serve dinner ?

Why is Part-Of-Speech Tagging Hard?

- Words may be ambiguous in different ways:
  - A word may have multiple meanings as the same part-of-speech
    - *file* – **noun**, a folder for storing papers
    - *file* – **noun**, instrument for smoothing rough edges
  - A word may function as multiple parts-of-speech
    - a *round* table: **adjective**
    - a *round* of applause: **noun**
    - to *round* out your interests: **verb**
    - to work the year *round*: **adverb**

Why is Part-Of-Speech Tagging Needed?

- May be useful to know what function the word plays, instead of depending on the word itself.
- Internally, next higher levels of NL Processing:
  - Phrase Bracketing
    - Can write regexps like (Det) Adj* N+ over the output for phrases, etc.
  - Parsing
    - As input to or to speed up a full parser
    - If you know the tag, you can back off to it in other tasks
  - Semantics
- Applications that use POS tagging:
  - Speech synthesis - Text-to-speech (how do we pronounce “lead”?)
  - Information retrieval — stemming, selection of high-content words
  - Word-sense disambiguation
  - Machine Translation
  - and others

Overview of Approaches

- Rule-based Approach
  - Simple and doesn’t require a tagged corpus, but not as accurate as other approaches
- Stochastic Approach
  - Refers to any approach which incorporates frequencies or probabilities
  - Requires a tagged corpus to learn frequencies
  - N-gram taggers and Naïve Bayes taggers
  - Hidden Markov Model (HMM) taggers
  - . . .
- Other Issues: unknown words and evaluation

The Problem

- Words often have more than one word class: another example is the word *this*
  - *This* is a nice day = PRP
  - *This* day is nice = DT
  - You can go *this* far = RB

Word Class Ambiguity (in the Brown Corpus)

- Unambiguous (1 tag): 35,340
- Ambiguous (2-7 tags): 4,100

2 tags	3,760
3 tags	264
4 tags	61
5 tags	12
6 tags	2
7 tags	1

(Derose, 1988)

Rule-Based Tagging

- Uses a dictionary that gives possible tags for words
- Basic algorithm
  - Assign all possible tags to words
  - Remove tags according to set of rules of type:
    - Example rule:
      - if word+1 is an adj, adv, or quantifier and the following is a sentence boundary and word-1 is not a verb like "consider" then eliminate non-adv else eliminate adv.
  - Typically more than 1000 hand-written rules, but may be machine-learned
- This approach not is serious use

N-gram Approach

- N-gram approach to probabilistic POS tagging:
  - calculates the probability of a given sequence of tags occurring for a sequence of words
  - the best tag for a given word is determined by the (already calculated) probability that it occurs with the n previous tags
  - may be bi-gram, tri-gram, etc

wordn-1	…	word-2	word-1	word
tagn-1	…	tag-2	tag-1	??

- Presented here as an introduction to HMM tagging
  - And given in more detail in the NLTK
  - In practice, bigram and trigram probabilities have the problem that the combinations of words are sparse in the corpus
  - Combine the taggers with a backoff approach

N-gram Tagging

- Initialize a tagger by learning probabilities from a tagged corpus

wordn_{-1}	…	word_{-2}	word_{-1}	word
tagn_{-1}	…	tag_{-2}	tag_{-1}	??

  - Probability that the sequence … tag_{-2} tag_{-1} word gives tag XX
  - Note that initial sequences will include a start marker as part of the sequence
- Use the tagger to tag word sequences (usually of length 2-3) with unknown tags
  - Sequence through the words:
    - To determine the POS tag for the next word, use the previous n-1 tags and the word to look up probabilities and use the highest probability tag

Need Longer Sequence Classification

- A more comprehensive approach to tagging considers the entire sequence of words
  - *Secretariat is expected to race tomorrow*
- What is the best sequence of tags which corresponds to this sequence of observations?
- Probabilistic view:
  - Consider all possible sequences of tags
  - Out of this universe of sequences, choose the tag sequence which is most probable given the observation sequence of n words w1…wn.

Thanks to Jim Martin's online class slides for the examples and equation typesetting in this section on HMM's.

Road to HMMs

- We want, out of all sequences of n tags $t_1 \ldots t_n$ the single tag sequence such that $P(t_1 \ldots t_n | w_1 \ldots w_n)$ is highest.
  - i.e. the probability of the tag sequence $t_1 \ldots t_n$ given the word sequence $w_1 \ldots w_n$

*

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

- Hat ^ means "our estimate of the best one"
- Argmax$_x$ f(x) means "the x such that f(x) is maximized"
  - i.e. find the tag sequence that maximizes the probability

Road to HMMs

- This equation is guaranteed to give us the best tag sequence

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n | w_1^n)$$

- But how to make it operational? How to compute this value?
- Intuition of Bayesian classification:
  - Use Bayes rule to transform into a set of other probabilities that are easier to compute

Thomas Bayes 1701 - 1761

Using Bayes Rule

- Bayes rule:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}$$

- Apply Bayes Rule:

$$\hat{t}_1^n = \operatorname*{argmax}_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)}$$

- Note that this is using the conditional probability, given a tag, what is the most likely word with that tag.
  - Eliminate denominator as it is the same for every sequence

$$\hat{t}_1^n = \operatorname*{argmax}_{t_1^n} P(w_1^n|t_1^n)P(t_1^n)$$

Likelihood and Prior

- Further simplify

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} \overbrace{P(w_1^n|t_1^n)}^{\text{likelihood}} \overbrace{P(t_1^n)}^{\text{prior}}$$

- Likelihood: assume that the probability of the word depends only on its tag

$$P(w_1^n|t_1^n) \approx \prod_{i=1}^{n} P(w_i|t_i)$$

- Prior: use the bigram assumption that the tag only depends on the previous tag

$$P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1})$$

$$\hat{t}_1^n = \underset{t_1^n}{\operatorname{argmax}} P(t_1^n|w_1^n) \approx \underset{t_1^n}{\operatorname{argmax}} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1})$$

Two Sets of Probabilities (1)

- Tag transition probabilities $p(t_i|t_{i-1})$ (priors)
  - Determiners likely to precede adjs and nouns
    - That/DT flight/NN
    - The/DT yellow/JJ hat/NN
    - So we expect P(NN|DT) and P(JJ|DT) to be high
  - Compute P(NN|DT) by counting in a labeled corpus:

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

Count of DT NN sequence

$$P(NN|DT) = \frac{C(DT, NN)}{C(DT)} = \frac{56,509}{116,454} = .49$$

Two Sets of Probabilities (2)

- Word likelihood probabilities p(w_i|t_i)
  - VBZ (3sg Pres verb) likely to be "is"
  - Compute P(is|VBZ) by counting in a labeled corpus:

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

Count of is tagged with VBZ

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10,073}{21,627} = .47$$

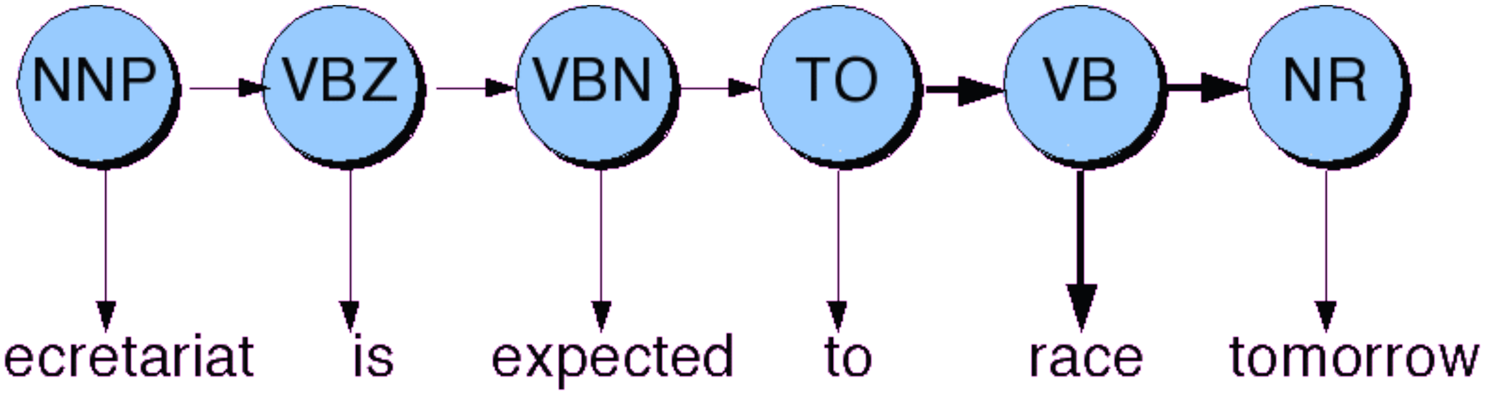
An Example: the word “race”

- The word “race” can occur as a verb or as a noun:
  - Secretariat/NNP is/VBZ expected/VBN to/TO **race**/VB tomorrow/NR
  - People/NNS continue/VB to/TO inquire/VB the/DT reason/NN for/IN the/DT **race**/NN for/IN outer/JJ space/NN
- How do we pick the right tag?

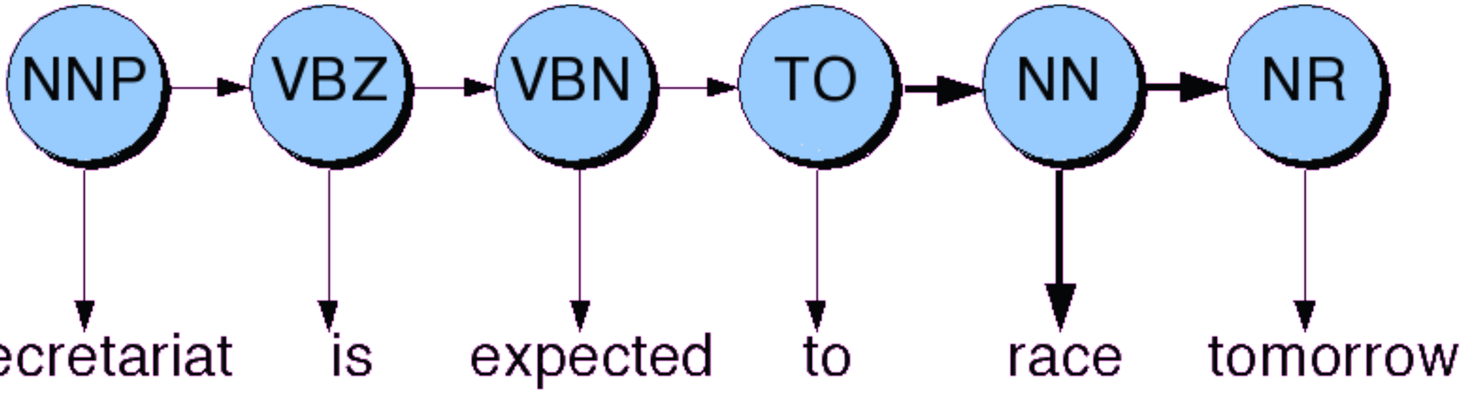
Disambiguating “race”

Which tag sequence is most likely?

(a)



(b)



Example

- *The equations only differ in "to race tomorrow"*
- P(NN|TO) = .00047
- P(VB|TO) = .83
- P(race|NN) = .00057
- P(race|VB) = .00012
- P(NR|VB) = .0027
- P(NR|NN) = .0012

The tag transition probabilities P(NN|TO) and P(VB|TO)

Lexical likelihoods from the Brown corpus for 'race' given a POS tag NN or VB.

Tag sequence probability for the likelihood of an adverb occurring given the previous tag verb or noun

- P(VB|TO)P(NR|VB)P(race|VB) = .00000027
- P(NN|TO)P(NR|NN)P(race|NN)=.00000000032
- *So we (correctly) choose the verb tag.*

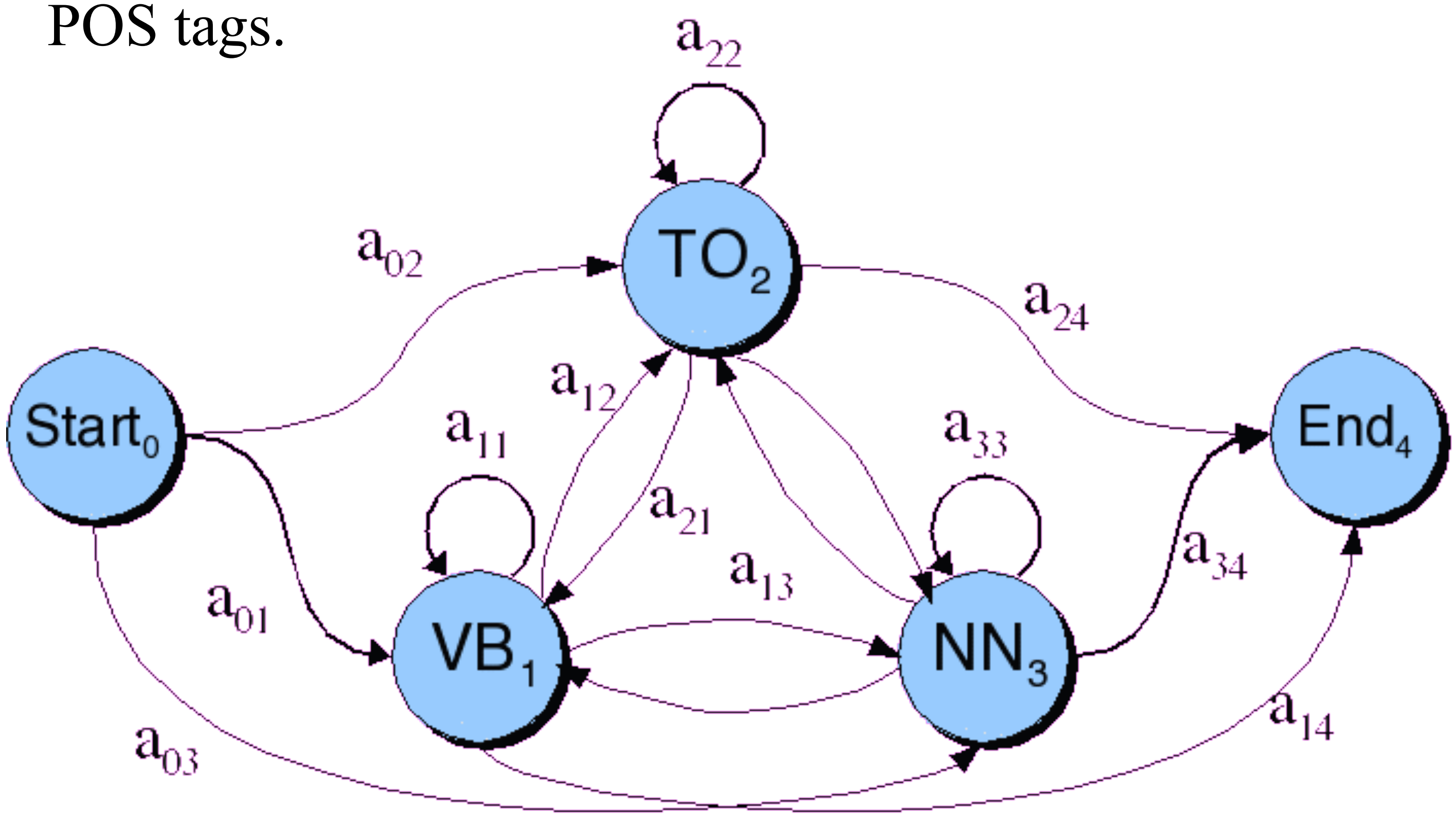
In-class Exercise

Hidden Markov Models

- What we' ve described with these two kinds of probabilities is a Hidden Markov Model
  - The Markov Model is the sequence of words and the hidden states are the POS tags for each word.
- When we evaluated the probabilities by hand for a sentence, we could pick the optimum tag sequence
- But in general, we need an optimization algorithm to most efficiently pick the best tag sequence without computing all possible combinations of probabilities

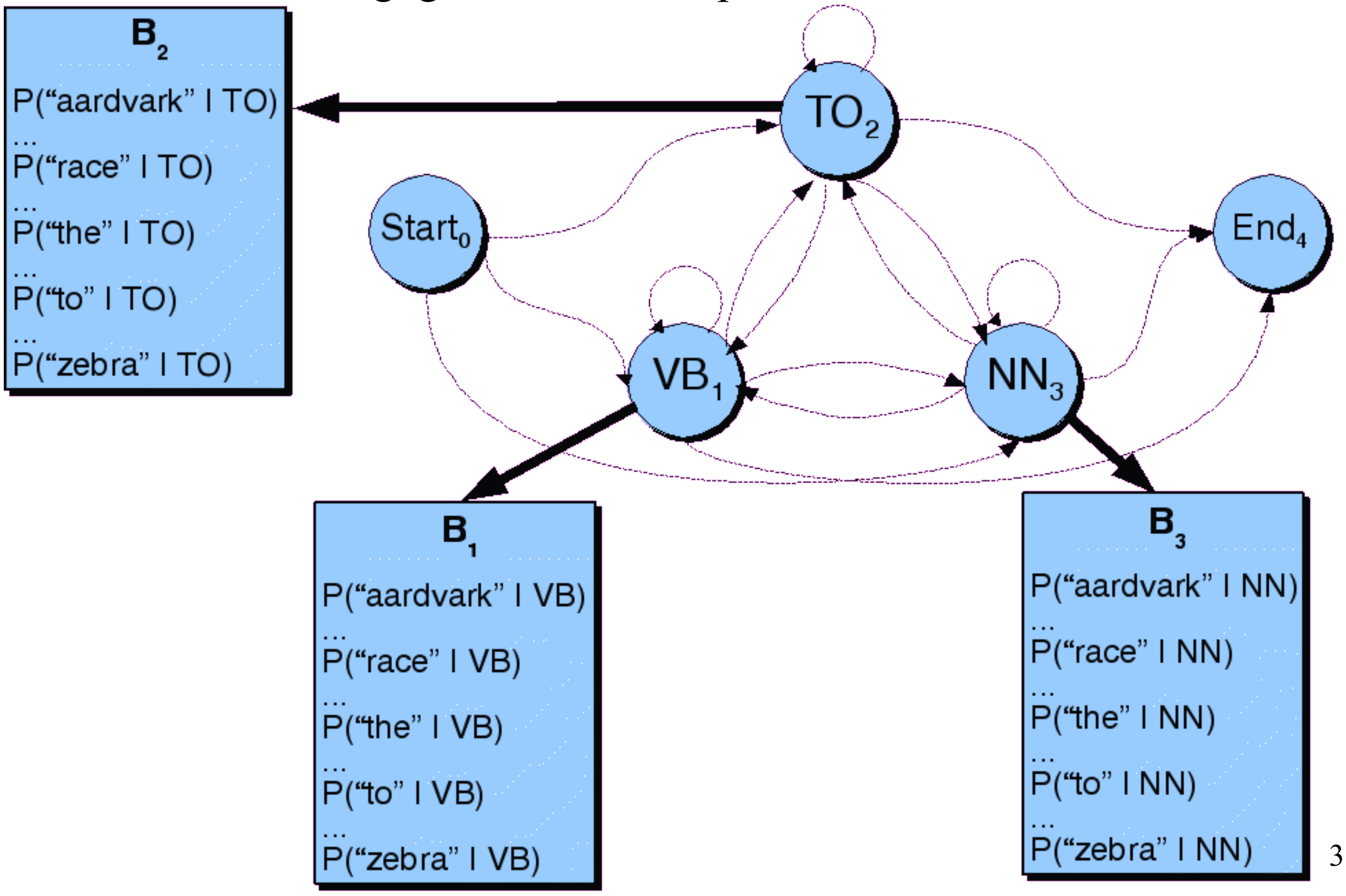
Tag Transition Probabilities for an HMM

- The HMM hidden states can be represented in a graph where the edges are the transition probabilities between POS tags.



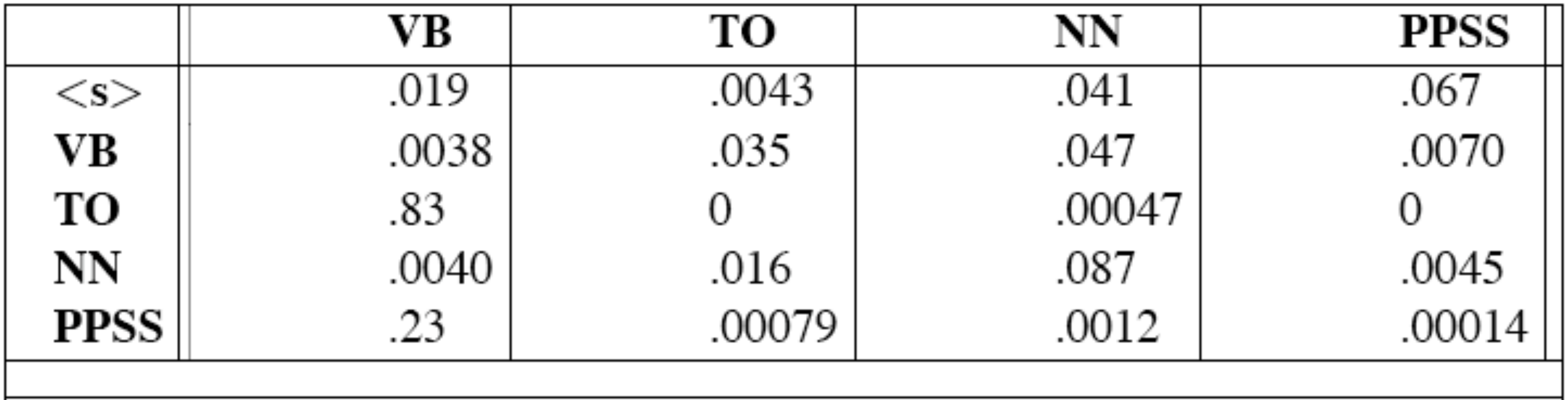
Observation likelihoods for POS HMM

- For each POS tag, give words with probabilities



The A matrix for the POS HMM

- Example of tag transition probabilities represented in a matrix, usually called the A matrix in an HMM:
  - The probability that VB follows <s> is .019, …

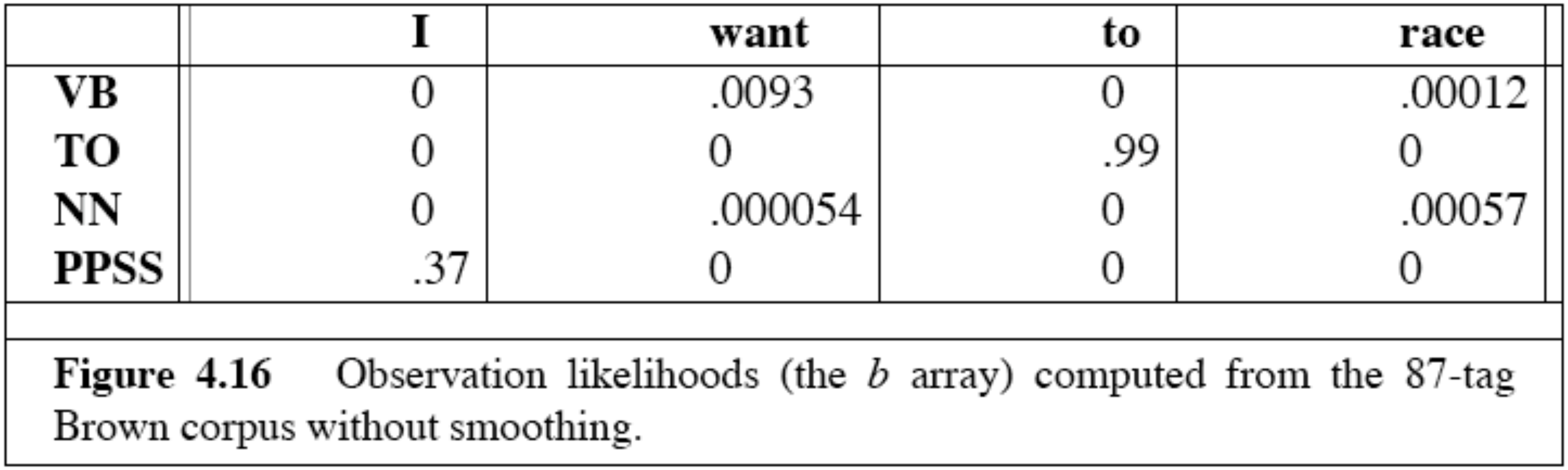


	VB	TO	NN	PPSS
<s>	.019	.0043	.041	.067
VB	.0038	.035	.047	.0070
TO	.83	0	.00047	0
NN	.0040	.016	.087	.0045
PPSS	.23	.00079	.0012	.00014

Figure 4.15 Tag transition probabilities (the *a* array, $p(t_i|t_{i-1})$ computed from the 87-tag Brown corpus without smoothing. The rows are labeled with the conditioning event; thus $P(PPSS|VB)$ is .0070. The symbol <s> is the start-of-sentence symbol.

The B matrix for the POS HMM

- Word likelihood probabilities are represented in a matrix, where for each tag, we show the probability that a word has that tag



	I	**want**	**to**	**race**
VB	0	.0093	0	.00012
TO	0	0	.99	0
NN	0	.000054	0	.00057
PPSS	.37	0	0	0

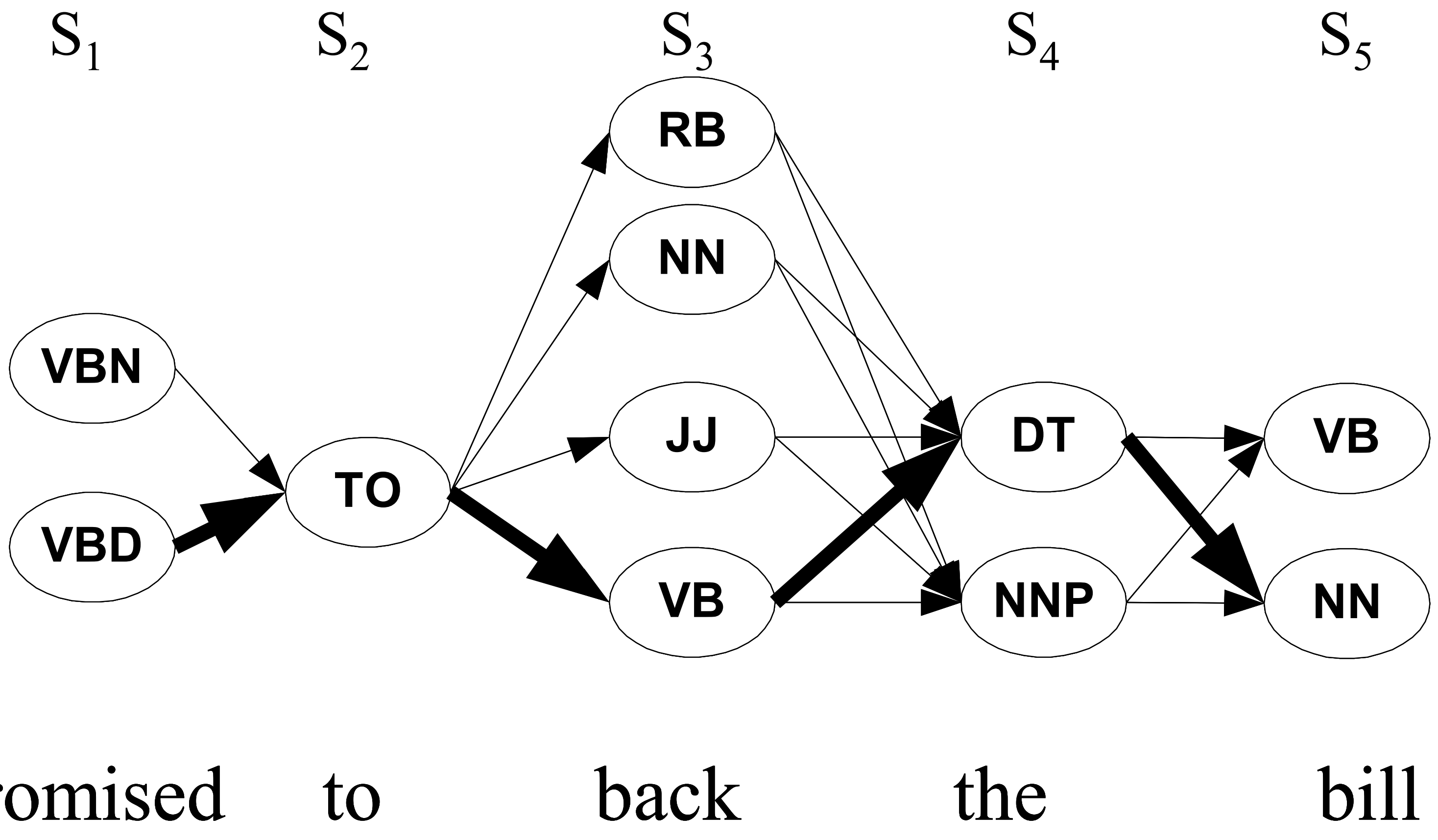
Figure 4.16 Observation likelihoods (the b array) computed from the 87-tag Brown corpus without smoothing.

Using HMMs for POS tagging

- From the tagged corpus, create a tagger by computing the two matrices of probabilities, A and B
  - Straightforward for bigram HMM
  - For higher-order HMMs, efficiently compute matrix by the forward-backward algorithm
- To apply the HMM tagger to unseen text, we must find the best sequence of transitions
  - Given a sequence of words, find the sequence of states (POS tags) with the highest probabilities along the path
  - This task is sometimes called “decoding”
  - Use the Viterbi algorithm

Viterbi intuition: we are looking for the best ‘path’

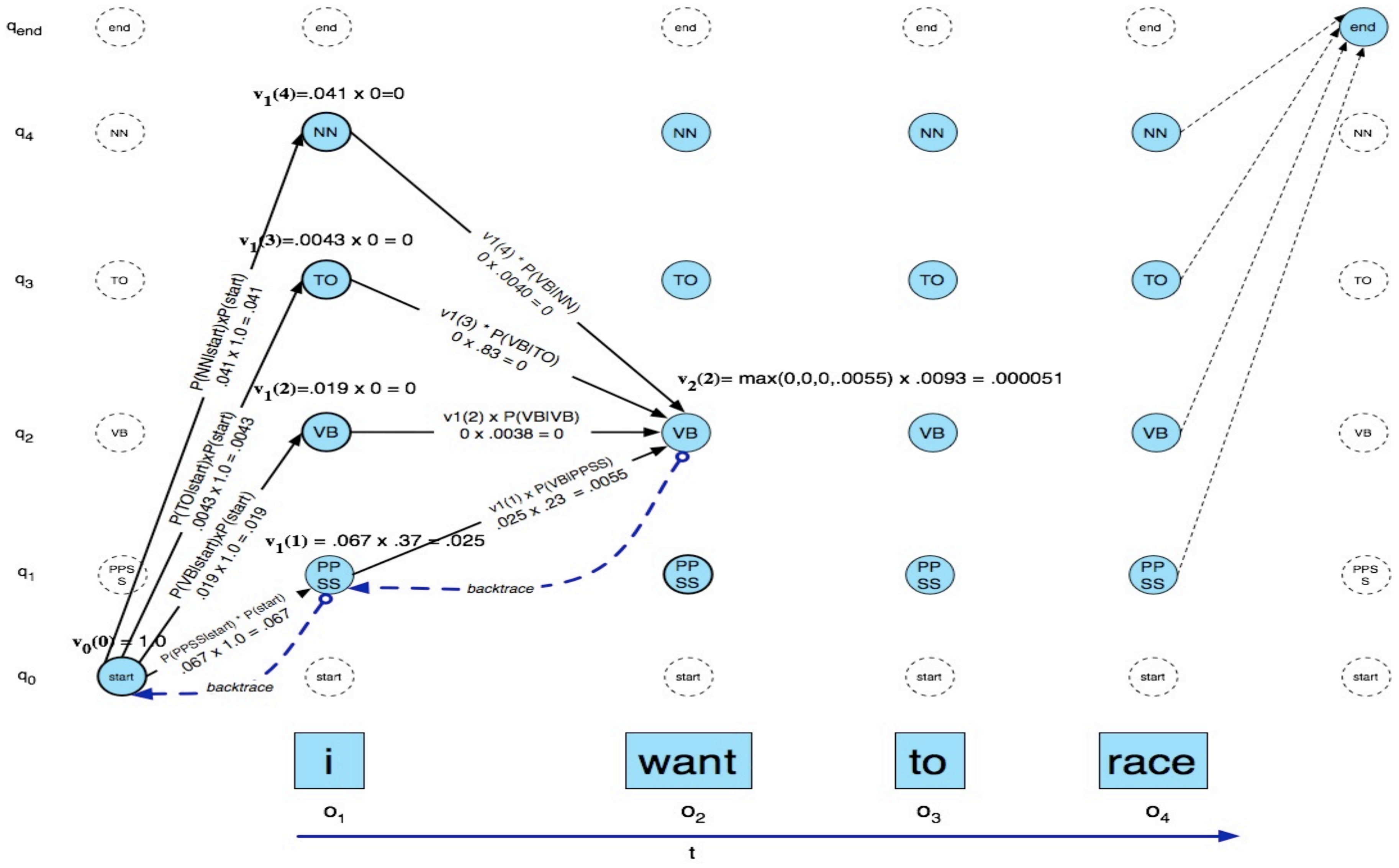
Each word has states representing the possible POS tags:



Viterbi example

Each pair of tags labeled with an edge giving transition probability

Each tag in a state labeled with a Viterbi value giving max over states in previous word of:
(its viterbi value * transition prob * word likelihood), representing "best path to this node"



q_{end}
q_4
q_3
q_2
q_1
q_0
end
NN
TO
VB
PPSS
start
$v_1(4)$=.041 x 0=0
$v_1(3)$=.0043 x 0 = 0
$v_1(2)$=.019 x 0 = 0
$v_1(1)$ = .067 x .37 = .025
$v_0(0)$ = 1.0
P(NN|start)xP(start) .041 x 1.0 = .041
P(TO|start)xP(start) .0043 x 1.0 = .0043
P(VB|start)xP(start) .019 x 1.0 = .019
P(PPSS|start) * P(start) .067 x 1.0 = .067
v1(4) * P(VB|NN) 0 x .0040 = 0
v1(3) * P(VB|TO) 0 x .83 = 0
v1(2) x P(VB|VB) 0 x .0038 = 0
v1(1) x P(VB|PPSS) .025 x .23 = .0055
$v_2(2)$= max(0,0,0,.0055) x .0093 = .000051
backtrace
backtrace
i
want
to
race
o_1
o_2
o_3
o_4
t

Viterbi Algorithm sketch

- This algorithm fills in the elements of the array viterbi in the previous slide (cols are words, rows are states (POS tags))

```
function Viterbi
    for each state s, compute the initial column
        viterbi[s, 1] = A[0, s] * b[s, word1]
    for each word w from 2 to N (length of sequence)
        for each state s, compute the column for w
            viterbi[s, w] =
                    max over s' ( viterbi[s',w-1] * A[s',s] * B[s,w])
            <save back pointer to trace final path>
    return the trace of back pointers
```

where A is the matrix of state transitions

and B is the matrix of state/word likelihoods

Recall HMM

- So an HMM POS tagger computes the A matrix of tag transition probabilities and the B matrix of likelihood tag/word probabilities from a (training) corpus
- Then for each sentence that we want to tag, it uses the Viterbi algorithm to find the path of the best sequence of tags to fit that sentence.
- This is an example of a sequential classifier.

Evaluation: Is our POS tagger any good?

- Answer: we use a manually tagged corpus, which we will call the "Gold Standard"
  - We run our POS tagger on the gold standard and compare its predicted tags with the gold tags
  - We compute the accuracy (and other evaluation measures)
- Important: 100% is impossible even for human annotators.
  - We estimate humans can do POS tagging at about 98% accuracy.
  - Some tagging decisions are very subtle and hard to do:
    - 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
  - The "Gold Standard" will have human mistakes; humans are subject to fatigue, etc.

How can we improve our tagger?

- What are the main sources of information for our HMM POS tagger?
  - Knowledge of tags of neighboring words
  - Knowledge of word tag probabilities
    - *man* is rarely used as a verb….
- The latter proves the most useful, but the former also helps
- Unknown words can be a problem because we don’t have this information
- And we are not including information about the features of the words

Features of words

- Can do surprisingly well just looking at a word by itself:

– Word	the: the → DT (determiner)
– Lowercased word	Importantly: importantly → RB (adverb)
– Prefixes	unfathomable: un- → JJ (adjective)
– Suffixes	Importantly: -ly → RB tangential: -al → JJ
– Capitalization	Meridian: CAP → NNP (proper noun)
– Word shapes	35-year: d-x → JJ

- These properties can include information about the previous or the next word(s)
  - The word *be* appears to the left pretty → JJ
- But not information about tags of the previous or next words, unlike HMM

Feature-based Classifiers

- A feature-based classifier is an algorithm that will take a word and assign a POS tag based on features of the word in its context in the sentence.
- Many algorithms are used, just to name a few
  - Naïve Bayes
  - Maximum Entropy (MaxEnt)
  - Support Vector Machines (SVM)

  - We'll be covering lots more about classifiers later in the course.

Comparison of HMM and feature-based classifiers

- Recall that HMM (and n-gram) taggers are sequential classifiers that use the previous sequence of tags as information:

  wordn_{-1} … word_{-2} word_{-1} word

  tagn_{-1} … tag_{-2} tag_{-1} ??

  – In order from left to right, use information from previous tags (tag prior probabilities) and word (word likelihood probabilities) to predict the next tag in the sequence

- Instead a feature-based classifier is looking just at the word and properties/features of the surrounding words

  word_{-2} word_{-1} word word_{+1} word_{+2}

  ??

  – Assign a tag XX to the word

Overview of POS tagger Accuracies

- List produced by Chris Manning
- Rough accuracies: all words / unknown words

– Most freq tag:	~90% / ~50%
– Trigram HMM:	~95% / ~55%
• HMM with trigrams	
– Maxent P(t|w):	93.7% / 82.6%
• Feature based tagger	
– MEMM tagger:	96.9% / 86.9%
• Combines feature based and HMM tagger	
– Bidirectional dependencies:	97.2% / 90.0%
– Upper bound:	~98% (human agreement)

Most errors on unknown words

Development process for features

- The tagged data should be separated into a training set and a test set.
  - The tagger is trained on the training set and evaluated on the test set
    - May also hold out some data for development
  - Evaluation numbers are not prejudiced by the training set
- If our feature-based tagger has errors, then we improve the features.
  - Suppose we incorrectly tag *as* as IN in the phrase *as soon as*, when it should be RB:

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

  - We could fix this with a feature that include the next word.

POS taggers with online demos

- Many pages list downloadable taggers (and other resources) such as this page from the Stanford NLP group and George Dillon at U Washington
  - http://nlp.stanford.edu/software/tagger.shtml
  - http://faculty.washington.edu/dillon/GramResources/
- There are not too many on-line taggers available for demos, but here are some possibilities:
  - The Stanford online parser demo includes POS tags: http://nlp.stanford.edu:8080/parser/ http://nlp.stanford.edu:8080/corenlp/
  - Illinois (UIUC) tagger demo from the Cognitive Computation Group
  - http://cogcomp.cs.illinois.edu/demo/pos/?id=4 (colors!)

Conclusions

- Part of Speech tagging is a doable task with high performance results
- Contributes to many practical, real-world NLP applications and is now used as a pre-processing module in most systems
- Computational techniques learned at this level can be applied to NLP tasks at higher levels of language processing