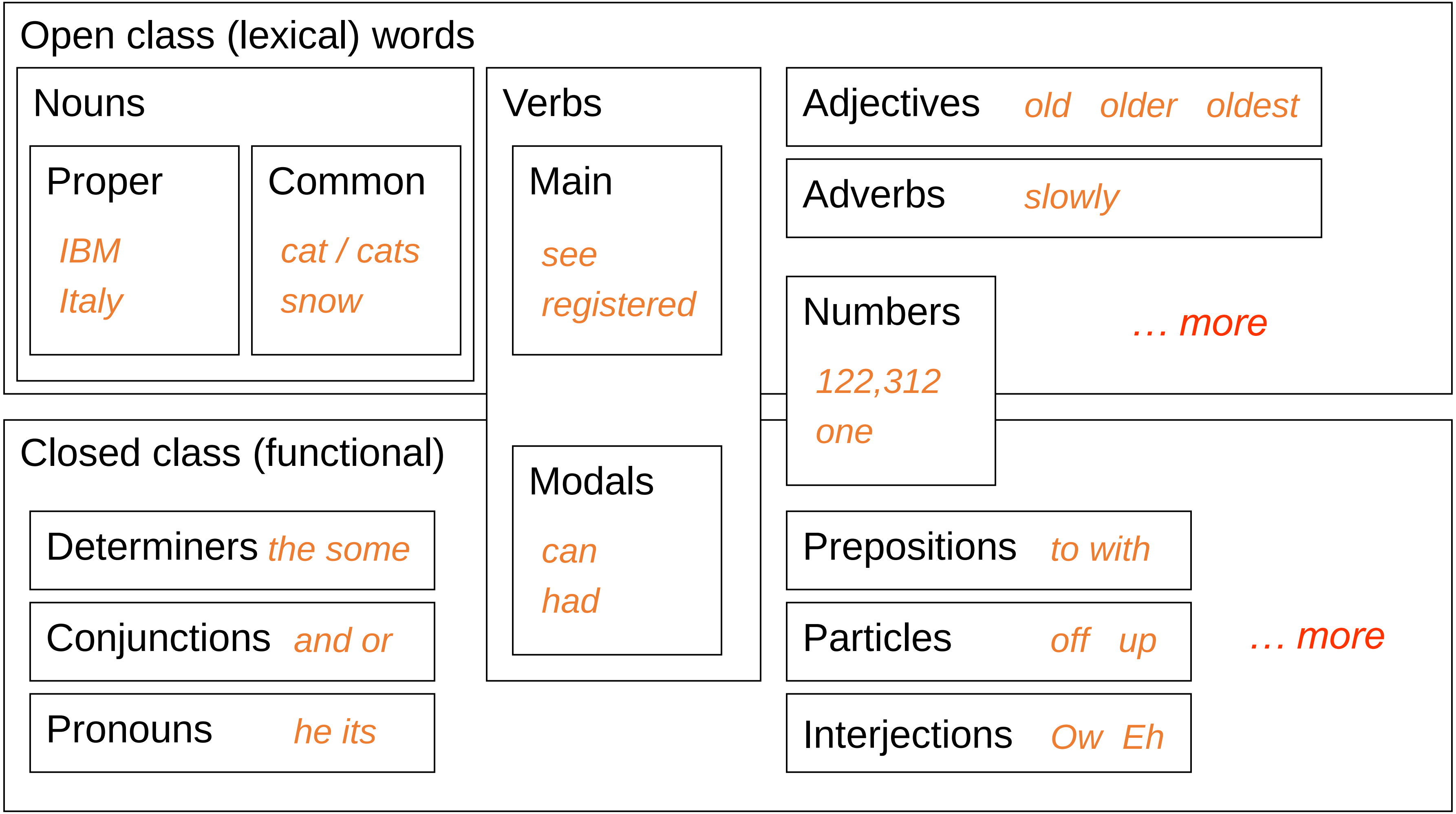
Part-of-speech tagging

Yuguang Zhang
CS 886: Topics in Natural Language Processing
University of Waterloo
Spring 2015

Parts of Speech

- Perhaps starting with Aristotle in the West (384–322 BCE), there was the idea of having parts of speech
  - a.k.a lexical categories, word classes, “tags”, POS
- It comes from Dionysius Thrax of Alexandria (c. 100 BCE) the idea that is still with us that there are 8 parts of speech
  - But actually his 8 aren’t exactly the ones we are taught today
    - Thrax: noun, verb, article, adverb, preposition, conjunction, participle, pronoun
    - School grammar: noun, verb, adjective, adverb, preposition, conjunction, pronoun, interjection


Open class (lexical) words
Nouns
Proper
IBM
Italy
Common
cat / cats
snow
Verbs
Main
see
registered
Modals
can
had
Adjectives old older oldest
Adverbs slowly
Numbers
122,312
one
… more
Closed class (functional)
Determiners the some
Conjunctions and or
Pronouns he its
Prepositions to with
Particles off up
Interjections Ow Eh
… more

Open vs. Closed classes

- Open vs. Closed classes
  - Closed:
    - determiners: ***a, an, the***
    - pronouns: ***she, he, I***
    - prepositions: ***on, under, over, near, by, …***
    - Why “closed”?
  - Open:
    - Nouns, Verbs, Adjectives, Adverbs.

POS Tagging

- Words often have more than one POS: *back*
  - The *back* door = JJ
  - On my *back* = NN
  - Win the voters *back* = RB
  - Promised to *back* the bill = VB
- The POS tagging problem is to determine the POS tag for a particular instance of a word.

POS Tagging

- Input: Plays well with others
- Ambiguity: NNS/VBZ UH/JJ/NN/RB IN NNS
- Output: Plays/VBZ well/RB with/IN others/NNS
- Uses:
  - Text-to-speech (how do we pronounce "lead"?)
  - Can write regexps like (Det) Adj* N+ over the output for phrases, etc.
  - As input to or to speed up a full parser
  - If you know the tag, you can back off to it in other tasks

Penn Treebank POS tags

POS tagging performance

- How many tags are correct? (Tag accuracy)
  - About 97% currently
  - But baseline is already 90%
    - Baseline is performance of stupidest possible method
      - Tag every word with its most frequent tag
      - Tag unknown words as nouns
  - Partly easy because
    - Many words are unambiguous
    - You get points for them (*the*, *a*, etc.) and for punctuation marks!

Deciding on the correct part of speech can be difficult even for people

- 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

How difficult is POS tagging?

- About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech
- But they tend to be very common words. E.g., *that*
  - I know *that* he is honest = IN (Preposition)
  - Yes, *that* play was nice = DT (Determiner)
  - You can’t go *that* far = RB (Adverb)
- 40% of the word tokens are ambiguous

A Maximum Entropy Model for POS Tagging

Adwait Ratnaparkhi

Sources of information

- Large annotated corpora for learning probability distributions
  - *man* is rarely used as a verb….
- Word context
  - Bill saw that man yesterday
  - NNP NN DT NN NN
  - VB VB(D) IN VB NN

Probability model

- $p(h,t) \quad = \pi\mu \prod_{j=1}^{k} a_j^{f_j(h,t)}$
- h history
- t tag
- f_j features
- μ, a_j model parameters

- $h_i = \{w_i, w_{i+1}, w_{i+2}, w_{i-1}, w_{i-2}, t_{i-1}, t_{i-2}\}$
- $p(h,t)$ is determined by the a_j such that $f_j(h,t)=1$
- $\{\mu, a_1, a_2, \ldots, a_k\}$ are chosen to maximize the likelihood of training data

Other uses for the Maxent model

- You can use a maxent classifier whenever you want to assign data points to one of a number of classes:
  - Sentence boundary detection (Mikheev 2000)
    - Is a period end of sentence or abbreviation?
  - Sentiment analysis (Pang and Lee 2002)
    - Word unigrams, bigrams, POS counts, …
  - Machine translation (Pang and Lee 2002)
  - Prepositional phrase attachment (Ratnaparkhi 1998)
    - Attach to verb or noun? Features of head noun, preposition, etc.
  - Parsing decisions in general (Ratnaparkhi 1997; Johnson et al. 1999, etc.)

An Example

Word:	The	stories	about	well-heeled	communities	and	developers
Tag	DT	NNS	IN	JJ	NNS	CC	NNS
Position	1	2	3	4	5	6	7

Example - Common Word

$w_i = \texttt{about}$ & $t_i = \texttt{IN}$
$w_{i-1} = \texttt{stories}$ & $t_i = \texttt{IN}$
$w_{i-2} = \texttt{the}$ & $t_i = \texttt{IN}$
$w_{i+1} = \texttt{well-heeled}$ & $t_i = \texttt{IN}$
$w_{i+2} = \texttt{communities}$ & $t_i = \texttt{IN}$
$t_{i-1} = \texttt{NNS}$ & $t_i = \texttt{IN}$
$t_{i-2}t_{i-1} = \texttt{DT NNS}$ & $t_i = \texttt{IN}$

Condition	Features	
w_i is not rare	$w_i = X$	& $t_i = T$
w_i is rare	X is prefix of w_i, $\lvert X \rvert \leq 4$	& $t_i = T$
	X is suffix of w_i, $\lvert X \rvert \leq 4$	& $t_i = T$
	w_i contains number	& $t_i = T$
	w_i contains uppercase character	& $t_i = T$
	w_i contains hyphen	& $t_i = T$
$\forall\, w_i$	$t_{i-1} = X$	& $t_i = T$
	$t_{i-2}t_{i-1} = XY$	& $t_i = T$
	$w_{i-1} = X$	& $t_i = T$
	$w_{i-2} = X$	& $t_i = T$
	$w_{i+1} = X$	& $t_i = T$
	$w_{i+2} = X$	& $t_i = T$

Example – Rare Word

$w_{i-1} =$ `about`	$\& \ t_i =$ `JJ`
$w_{i-2} =$ `stories`	$\& \ t_i =$ `JJ`
$w_{i+1} =$ `communities`	$\& \ t_i =$ `JJ`
$w_{i+2} =$ `and`	$\& \ t_i =$ `JJ`
$t_{i-1} =$ `IN`	$\& \ t_i =$ `JJ`
$t_{i-2}t_{i-1} =$ `NNS IN`	$\& \ t_i =$ `JJ`
prefix(w_i)=`w`	$\& \ t_i =$ `JJ`
prefix(w_i)=`we`	$\& \ t_i =$ `JJ`
prefix(w_i)=`wel`	$\& \ t_i =$ `JJ`
prefix(w_i)=`well`	$\& \ t_i =$ `JJ`
suffix(w_i)=`d`	$\& \ t_i =$ `JJ`
suffix(w_i)=`ed`	$\& \ t_i =$ `JJ`
suffix(w_i)=`led`	$\& \ t_i =$ `JJ`
suffix(w_i)=`eled`	$\& \ t_i =$ `JJ`
w_i contains hyphen	$\& \ t_i =$ `JJ`

Condition	Features	
w_i is not rare	$w_i = X$	$\& \ t_i = T$
w_i is rare	X is prefix of w_i, $\|X\| \leq 4$	$\& \ t_i = T$
	X is suffix of w_i, $\|X\| \leq 4$	$\& \ t_i = T$
	w_i contains number	$\& \ t_i = T$
	w_i contains uppercase character	$\& \ t_i = T$
	w_i contains hyphen	$\& \ t_i = T$
$\forall \ w_i$	$t_{i-1} = X$	$\& \ t_i = T$
	$t_{i-2}t_{i-1} = XY$	$\& \ t_i = T$
	$w_{i-1} = X$	$\& \ t_i = T$
	$w_{i-2} = X$	$\& \ t_i = T$
	$w_{i+1} = X$	$\& \ t_i = T$
	$w_{i+2} = X$	$\& \ t_i = T$

Testing the model

- Wall St. Journal data
- Training set to train the statistical model
- Development set to tune parameters and decide on the best model
- Test set distinct from development set gives an estimate of error rate on real data

DataSet	Sentences	Words	Unknown Words
Training	40000	962687	
Development	8000	192826	6107
Test	5485	133805	3546

Procedure

- test corpus tagged one sentence at a time
- a modified beam search through possible tag sequences for a sentence
  - tag sequence with the highest probability selected
- O(NTAB) – running time with parameter estimation
  - B – beam size set to 5
  - N – training set size
  - T – number of allowable tags
  - A – average number of active features for an event *(h, t)*

Performance summary

		Total Word Accuracy	Unknown Word Accuracy	Sentence Accuracy
Development Set	Baseline with Tag Dictionary	96.43	86.23	47.55
	Baseline without Tag Dictionary	96.31	86.28	47.38
Test Set	Specialized Model	**96.63**	85.56	47.51

Specialized model for problematic words

Word	Correct Tag	Model's Tag	Frequency
about	RB	IN	393
that	DT	IN	389
more	RBR	JJR	221
up	IN	RB	187
that	WDT	IN	184
as	RB	IN	176
up	IN	RP	176
more	JJR	RBR	175
that	IN	WDT	159
about	IN	RB	144
that	IN	DT	127
out	RP	IN	126
that	DT	WDT	123
much	JJ	RB	118
yen	NN	NNS	117
chief	NN	JJ	116
up	RP	IN	114
ago	IN	RB	112
much	RB	JJ	111
out	IN	RP	109

Word	# Baseline Model Errors	# Specialized Model Errors
that	246	207
up	186	169
about	110	120
out	104	97
more	88	89
down	81	84
off	73	78
as	50	38
much	47	40
chief	46	47
in	39	39
executive	37	33
most	23	34
ago	22	18
yen	18	17

Overview: POS Tagging Accuracies

- Rough accuracies:
  - Most freq tag: ~90%
  - Trigram HMM: ~95%
  - Maxent P(t|w): 96.6%
  - TnT (HMM++): 96.2%
  - MEMM tagger: 96.9%
  - Bidirectional dependencies: 97.2%
  - Upper bound: ~98% (human agreement)

Feature-rich part-of-speech tagging with a cyclic dependency network

Toutanova et al.

How to solve this?

- Left to right factors do not always suffice

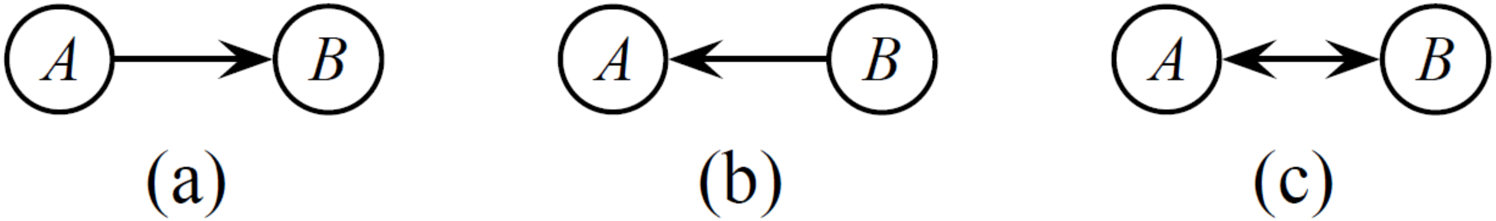
MD VB TO DT NN
Will go to the store

- The TO tag is most often preceded by noun, rarely a modal verb

MD
NN TO VB
Will to fight

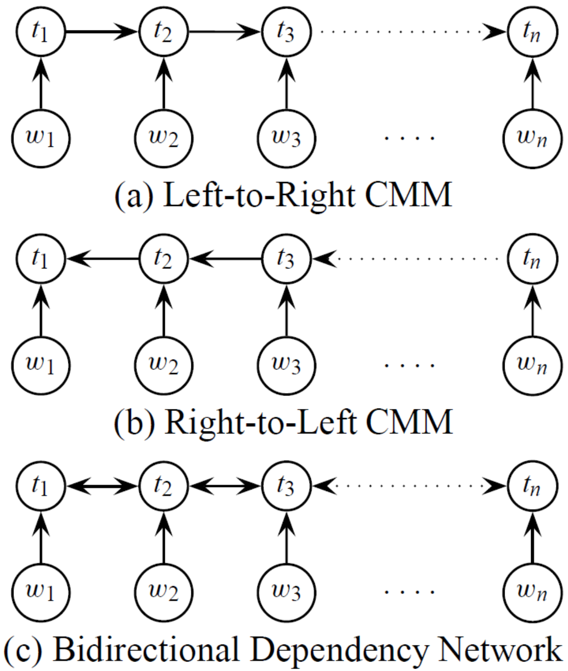
- $P(t_0|t_{-1})$ does not capture this, but $P(t_{-1}=NN|t_0=TO)$ does

Bayesian dependency networks



a) $P(A)P(B|A)$
b) $P(A|B)P(B)$
c) bidirectional net with models of $P(A|B)$ and $P(B|A)$

Dependency networks



$$p(t, w) = \prod_i P(\,?\,)$$

a) $P(t_i \mid t_{i-1}, w_i)$

b) $P(t_{i-1} \mid t_i, w_i)$

c) $P(t_i \mid t_{i-1}, t_{i+1}, w_i)$

Inference for linear dependency networks

```
function bestScore()
    return bestScoreSub(n + 2, ⟨end, end, end⟩);

function bestScoreSub(i + 1, ⟨t_{i−1}, t_i, t_{i+1}⟩)
    % memoization
    if (cached(i + 1, ⟨t_{i−1}, t_i, t_{i+1}⟩))
        return cache(i + 1, ⟨t_{i−1}, t_i, t_{i+1}⟩);
    % left boundary case
    if (i = −1)
        if (⟨t_{i−1}, t_i, t_{i+1}⟩ == ⟨start, start, start⟩)
            return 1;
        else
            return 0;
    % recursive case
    return max_{t_{i−2}} bestScoreSub(i, ⟨t_{i−2}, t_{i−1}, t_i⟩)×
                          P(t_i|t_{i−1}, t_{i+1}, w_i);
```

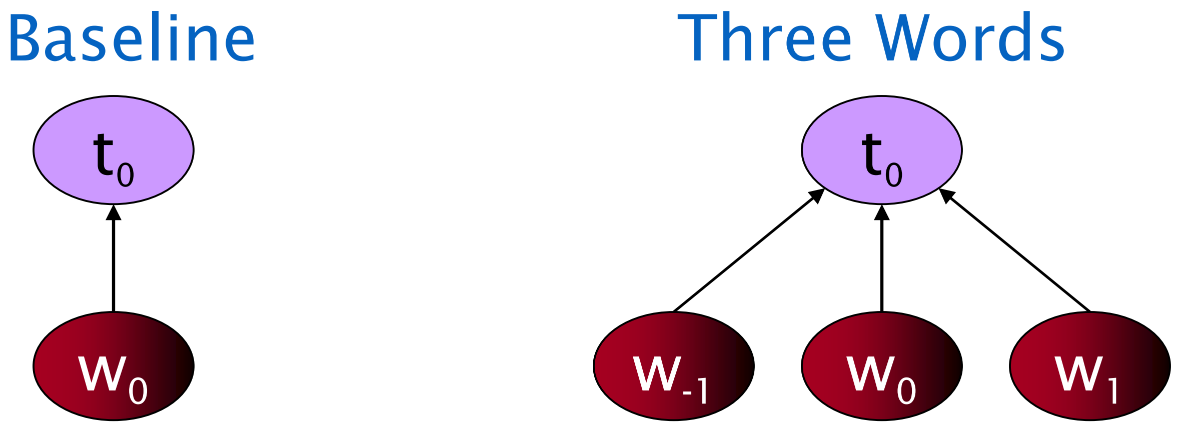
- Modified Viterbi algorithm to find the optimal sequence of tags
- Start from the last tag
- Multiply
  - best score for previous tag and
  - probability of current tag given word and surrounding tags

Directionality experiments

CMM performance with tags alone gives token accuracies of

- L: 95.79%
- R: 95.14%
- L+R: 96.57%
- LR: 96.55%
- L+LL+LR+RR+R: 96.92%
  - templates for TAGS in 3W+ TAGS

Lexicalization experiments



Model	Features	Sentence Accuracy	Token Accuracy	Unknown Accuracy
BASELINE	6,501	1.63%	60.16%	82.98%
3W	239,767	48.27%	96.57%	86.78%
3W+TAGS	263,160	53.83%	97.02%	88.05%
BEST	460,552	55.31%	97.15%	88.61%

Unknown word features

- Crude company name detector
  - Capitalized words followed within 3 words by *Co.*, *Inc.*, etc
- Minor:
  - allcaps
  - conjunction of allcaps and digits eg *CFC-12*
  - Prefixes and suffixes of length up to 10