# CS 294-5: Statistical **Natural Language Processing**



POS Tagging II Lecture 8: 9/26/05

## Recap: POS Ambiguity

Words are syntactically ambiguous:

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

- Two sources of information:
  - Clues from the input (current word, next word, capitalization, suffixes, word shape)
  - Clues from adjacent hidden labels (connectivity)
  - What of this could HMMs capture?
- Remember: POS sequence models will be the basis of information extraction methods later

### Recap: Accuracies

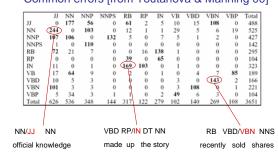
Most errors

on unknown words

- Roadmap of (known / unknown) accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% /~55%
  - Maxent P(t|w): 93.7% / 82.6% ■ TnT (HMM++): 96.2% / 86.0%
  - 96.9% / 86.9% Maxent tagger: Cyclic tagger: 97.2% / 89.0%
  - Upper bound: ~98%

# Recap: Errors

Common errors [from Toutanova & Manning 00]



#### **Better Features**

- Can do surprisingly well just looking at a word by itself:
  - Word the: the  $\rightarrow$  DT
  - Lowercased word Importantly: importantly  $\rightarrow RB$ Prefixes unfathomable: un-  $\rightarrow$  JJ Suffixes Importantly:  $-ly \rightarrow RB$  Capitalization Meridian: CAP → NNP

35-year: d-x → JJ

- Then build a maxent (or whatever) model to predict tag
- Maxent P(t|w): 93.7% / 82.6%

Word shapes

# Sequence-Free Tagging?

- What about looking at a word and it's environment, but no sequence information?
  - Add in previous / next word
- Previous / next word shapes Occurrence pattern features
- [X: x X occurs] ..... (Inc.|Co.)
- Crude entity detection Phrasal verb in sentence?
- Conjunctions of these things
- All features except sequence: 96.6% / 86.8%
- Uses lots of features: > 200K
- Why isn't this the standard approach?

## **Maxent Taggers**

One step up: also condition on previous tags

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

- Train up P(ti|w,ti-1,ti-2) as a normal maxent problem, then use to score sequences
- This is referred to as a maxent tagger [Ratnaparkhi
- Beam search effective! (Why?)
- What's the advantage of beam size 1?

#### **Feature Templates**

We've been sloppy:

Features: <w<sub>0</sub>=future, t<sub>0</sub>=JJ>

Feature templates: <w<sub>0</sub>, t<sub>0</sub>>

In maxent taggers:

Can now add edge feature templates:

< t<sub>-1</sub>, t<sub>0</sub>>

< t<sub>-2</sub>, t<sub>-1</sub>, t<sub>0</sub>>

Also, mixed feature templates:

 $< t_{-1}, w_0, t_0 >$ 

### Decoding

- Decoding maxent taggers:
  - Just like decoding HMMs
  - Viterbi, beam search, posterior decoding
- Viterbi algorithm (HMMs):

$$\delta_i(s) = \underset{s'}{\arg\max} \frac{P(s|s')P(w_{i-1}|s')\delta_{i-1}(s')}{\delta_{i-1}(s')}$$

Viterbi algorithm (Maxent):

$$\delta_i(s) = \arg\max_{s'} \frac{P(s|s', \mathbf{w})}{\delta_{i-1}(s')}$$

### TBL Tagger

[Brill 95] presents a transformation-based tagger

Label the training set with most frequent tags

DT MD VBD VBD The can was rusted

Add transformation rules which reduce training mistakes

MD → NN : DT \_\_\_
VBD → VBN : VBD

- Stop when no transformations do sufficient good
- Does this remind anyone of anything?
- Probably the most widely used tagger (esp. outside NLP)
- ... but not the most accurate: 96.6% / 82.0 %

# TBL Tagger II

What gets learned? [from Brill 95]

	Change Tag				Change Tag		
ű	From	To	Condition	#	From	To	Condition
1	NN	VB	Previous tag is TO		NN	NNS	Has suffix -s
2	VBP	VB	One of the previous three tags is MD	2	NN	CD	Has character .
3	NN	VB	One of the previous two tags is MD	3	NN	111	Has character -
4	VB	NN	One of the previous two tags is DT	4	NN	VBN	Has suffix -ed
5	VBD	VBN	One of the previous three tags is VBZ	5	NN	VBG	Has suffix -ing
6	VBN	VBD	Previous tag is PRP	6	77	RB	Has suffix •ly
7	VBN	VBD	Previous tag is NNP	7	22	11	Adding suffix -ly results in a word.
8	VBD	VBN	Previous tag is VBD	8	NN	CD	The word \$ can appear to the left.
9	VBP	VB	Previous tag is TO	9	NN	11	Bas suffix -al
10	POS	VBZ	Previous tag is PRP	10	NN	VB	The word would can appear to the left
11	VB	VBP	Previous tag is NNS	11	NN	CD	Has character 0
12	VBD	VBN	One of previous three tags is VBP	12	NN	11	The word be can appear to the left.
13	IN	WDT	One of next two tags is VB	13	NNS	IJ	Has suffix -us
14	VBD	VBN	One of previous two tags is VB	14	NNS	VBZ	The word it can appear to the left.
15	VB	VBP	Previous tag is PRP	15	NN	11)	Has suffix -ble
16	EN	WDT	Next tag is VBZ	16	NN	11	Has suffix -ic
17	IN	DT	Next tag is NN	17	NN	CD	Has character 1
18	11	NNP	Next tag is NNP	18	NNS	NN	Has suffix -ss
19	IN	WDT	Next tag is VBD	19	22	- 11	Deleting the prefix un- results in a wor
20	JJR	RBR	Next tag is JJ	20	NN	IJ	Has suffix -ive

# **EngCG Tagger**

- English constraint grammar tagger
  - [Tapanainen and Voutilainen 94]
  - Something else you should know about
  - Hand-written and knowledge driven
  - "Don't guess if you know" (general point about modeling more structure!)
  - Tag set doesn't make all of the hard distinctions as the standard tag set (e.g. JJ/NN)
  - They get stellar accuracies: 98.5% on their tag set
  - Linguistic representation matters...
  - ... but it's easier to win when you make up the rules

## **CRF Taggers**

- Newer, higher-powered discriminative sequence models
  - CRFs (also voted perceptrons, M3Ns)
  - Do not decompose training into independent local regions
  - Can be deathly slow to train require repeated inference on training set
- Differences tend not to be too important for POS tagging
- However: one issue worth knowing about in local models
  - "Label bias" and other explaining away effects
  - Maxent taggers' local scores can be near one without having both good "transitions" and "emissions"
  - This means that often evidence doesn't flow properly
  - Why isn't this a big deal for POS tagging?

#### **Domain Effects**

- Accuracies degrade outside of domain
  - Up to triple error rate
  - Usually make the most errors on the things you care about in the domain (e.g. protein names)
- Open questions
  - How to effectively exploit unlabeled data from a new domain (what could we gain?)
  - How to best incorporate domain lexica in a principled way (e.g. UMLS specialist lexicon, ontologies)

# **Unsupervised Tagging?**

- AKA part-of-speech induction
- Task:
  - Raw sentences in
  - Tagged sentences out
- Obvious thing to do:
  - Start with a (mostly) uniform HMM
  - Run EM
  - Inspect results

#### EM for HMMs: Quantities

Remember from last time:

$$\alpha_i(s) = P(w_0 \dots w_{i-1}, s_i)$$
  
= 
$$\sum_{s_{i-1}} P(s_i | s_{i-1}) P(w_{i-1} | s_{i-1}) \alpha_{i-1}(s_{i-1})$$

$$\beta_i(s) = P(w_i \dots w_n | s_i)$$
  
=  $\sum_{s_{i+1}} P(s_{i+1} | s_i) P(w_i | s_i) \beta_{i+1}(s_{i+1})$ 

Can calculate in O(s²n) time (why?)

## EM for HMMs: Process

• From these quantities, we can re-estimate transitions:

$$\mathsf{count}(s \to s') = \frac{\sum_i \alpha_i(s) P(s'|s) P(w_i|s) \beta_{i+1}(s')}{P(\mathbf{w})}$$

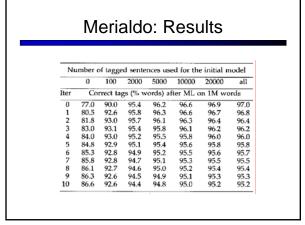
And emissions:

$$\mathrm{count}(w,s) = \frac{\sum_{i:w_i = w} \alpha_i(s)\beta_{i+1}(s)}{P(\mathbf{w})}$$

 If you don't get these formulas immediately, just think about hard EM instead, where were re-estimate from the Viterbi sequences

## Merialdo: Setup

- Some (discouraging) experiments [Merialdo 94]
- Setup
  - You know the set of allowable tags for each word
  - Fix k training examples to their true labels
    - Set P(w|t) on these examples
    - Set P(t|t<sub>-1</sub>,t<sub>-2</sub>) on these examples
  - Re-estimate with EM for n iterations
- Note: we know allowed tags but not frequencies



### So How to Fix It?

- Lots of progress in learning parts-of-speech
  - Distributional word clustering methods
  - Morphology diven models
  - Contrastive estimation
  - Other ideas!
- Stay tuned...