Part of Speech Tagging

Linguistics 409 · Computational Linguistics

Rice University

February 15, 2013

Coming up...

- Today: Parts of Speech and Tagsets
- Monday: Tagging

POS

8 (ish) traditional parts of speech

- Noun, verb, adjective, preposition, adverb, article, interjection, pronoun, conjunction, etc
- Called: parts-of-speech, lexical categories, word classes, morphological classes, lexical tags...
- Lots of debate within linguistics about the number, nature, and universality of these. We'll completely ignore this debate.

Examples

- **N** (noun) chair, bandwidth, pacing
- **V** (verb) study, debate, munch
- **ADJ** (adjective) purple, tall, ridiculous
- **ADV** (adverb) unfortunately, slowly
- **P** (preposition) of, by, to
- **PRO** (pronoun) I, me, mine
- **DET** (determiner) the, a, that, those

Semantic descriptions fail.

We typically try to define these (i.e. for children) semantically (see Schoolhouse Rock) but these descriptions ultimately fail.

Semantic descriptions fail.

A better description is a **functional** or **distributional** account:

You shall know a word by the company it keeps

Firth, J. R. (1957:11)

Ambiguity

entrance

Ambiguity

She tended to **entrance** her visitors.

Ambiguity

number

Ambiguity

My fingers grew **number** the higher we climbed.

Ambiguity

content

Ambiguity

Despite the cost, **content** was scarce.

Coca Corpus

- http://corpus.byu.edu/coca/
- http://corpus.byu.edu/bnc/help/pos_c7.asp

Brown Corpus

- http://archive.org/details/BrownCorpus
- http://www.comp.leeds.ac.uk/ccalas/tagsets/brown.html

For next time:

For next time:

1. Monday: **Part of speech tagging**
2. **Read:** J&M chapter 5 pp 139 - 149

Brown Corpus

- http://archive.org/details/BrownCorpus
- http://www.comp.leeds.ac.uk/ccalas/tagsets/brown.html

Two Basic Approaches to POS Tagging

- Rule-based tagging (e.g. ENGTWOL)
- Machine learning approaches (stochastic tagging)
  - HMM (Hidden Markov Model) tagging
  - MEMMs (Maximum Entropy Markov Models)

Rule-Based Tagging

- Start with a dictionary
- Write rules by hand to selectively remove tags
- Leaving the correct tag for each word.

Start with a dictionary...

a	<Indef> DET CENTRAL ART SG @DN>
aardvark	N NOM SG
abaci	<?> N NOM SG
aback	ADV
abacus	N NOM SG
abaft	<?> N NOM SG
abalone	<?> N NOM SG
abandon	*<Indef> N NOM SG* *<SVO> V PRES -SG3 VFIN @+FMAINV* *<SVO> V INF* *<SVO> V IMP VFIN @+FMAINV* *<SVO> V SUBJUNCTIVE VFIN @+FMAINV*
...	

Assign all possible tags to each word

			V-SUBJ		V-SUBJ
			V-INF		V-IMF
			V-IMP		V-IMP
	PCP2	PREP	V-PRES		V-PRES
PRON	V-PAST	TO	N-SQ	DET	N-SG
She	promised	to	back	the	bill

Write rules by hand to selectively remove tags

Eliminate PCP2 if V-PAST is an option when PCP2|V-PAST follows: <start> PRON

			V-SUBJ		V-SUBJ
			V-INF		V-IMF
			V-IMP		V-IMP
	PCP2	PREP	V-PRES		V-PRES
PRON	V-PAST	TO	N-SQ	DET	N-SG
She	promised	to	back	the	bill

Example: ENGTWOL (Stage 1)

- First Stage: Run words through FST morphological analyzer to get all parts of speech.
- Example: Pavlov had shown that salivation ...

Pavlov	PAVLOV N NOM SG PROPER
had	HAVE V PAST VFIN SVO HAVE PCP2 SVO
shown	SHOW PCP2 SVOO SVO SV
that	ADV PRON DEM SG DET CENTRAL DEM SG CS
salivation	N NOM SG

Example: ENGTWOL (Stage 2)

- Second Stage: Apply **negative** constraints (3,774 total)
- Example: adverbial *that* rule
- Eliminates all readings of *that* except the one in, e.g.:
- ‘It isn’t that odd’

```
Given input: that
If
(+1 A/ADV/QUANT) ;if next word is adj/adv/quantifier

(+2 SENT-LIM) ; following which is E-O-S

(NOT -1 SVOC/A) ; and the previous word is not a
                ; verb like 'consider' which
                ; allows adjective complements
                ; in 'I consider that odd'
Then eliminate non-ADV tags
Else eliminate ADV
```

Hidden Markov Model Tagging

- Using an HMM to do POS tagging is a special case of Bayesian inference.
- It is also related to the ‘noisy channel model’ that forms the basis for ASR, OCR and MT (as we’ll see later)

HMM Tagging

- Finds the highest probability sequence of tags given a sequence of words (though not directly!)
- Looks a wicked lot like minimum edit distance, e.g.
- Requires a training corpus/corpora and test corpus/corpora
- No probabilities for words not in corpus. (smoothing)
- Training corpus may be different from test corpus. (beware overfitting)
- We proceed by dynamic programming

HMM Tagging

- **Intuition:** Pick the most likely tag for each word.
- HMM Taggers choose tag sequence that maximizes this formula:

$$P(word|tag)xP(tag|previous\, n\, tags)$$

- Let $T = t_1, t_2, ..., t_n$
- Let $W = w_1, w_2, ..., w_n$
- Find POS tags that generate a sequence of words –look for most probable sequence of **hidden** tags T underlying the observed words W.

(Recall) Multiplication Rule

Multiplication Rule

In general: $P(e_1, e_2) = P(e_1)xP(e_2|e_1)$

We can rewrite the multiplication rule as a general definition for conditional probability of two events *e* and *f*:

Bayes' rule

$$P(e|f) = \frac{P(e,f)}{P(f)} = \frac{P(e)xP(f|e)}{P(f)}$$

Bigram HMM Tagger

- argmax P(T|W)
- argmax P(T)P(W|T)
- argmax $P(t_1 \ldots t_n)P(w_1 \ldots w_n | t_1 \ldots t_n)$
- argmax $[P(t_1)P(t_2|t_1) \ldots P(t_n|t_{n-1})][P(w_1|t_1)P(w_2|t_2) \ldots P(w_n|t_n)]$

To tag a single word:

$$t_i = argmax P(t_i | t_{i-1}) P(w_i | t_i)$$

Bigram HMM Tagger

$$t_i = argmax P(t_i|t_{i-1})P(w_i|t_i)$$

- How do we compute $P(t_i|t_{i-1})$?

$$\frac{c(t_{i-1}t_i)}{c(t_{i-1})}$$

- How do we compute $P(w_i|t_i)$?

$$\frac{c(w_i, t_i)}{c(t_i)}$$

- How do we compute the most probable tag sequence?

Viterbi

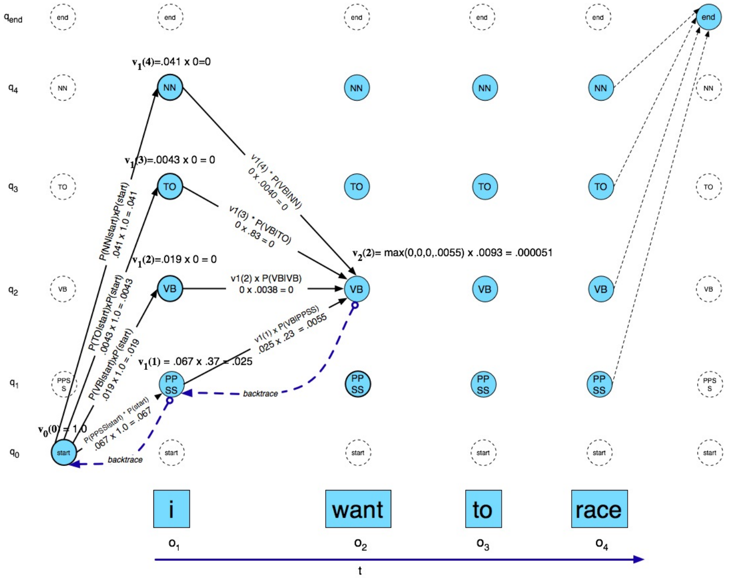
Viterbi Summary

- Create a matrix
  - Columns correspond to inputs
  - Rows correspond to possible states
- Sweep through the matrix in one pass filling the columns left to right using our transition probabilities and observations probabilities
- Key benefit of dynamic programming is that we need only store the MAX prob path to each cell (not the probabilities of all paths).

	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

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

JM Figure 5.18: example Viterbi lattice



Evaluation

	IN	JJ	NN	NNP	RB	VBD	VBN
IN	—	.2			.7		
JJ	.2	—	**3.3**	2.1	1.7	.2	**2.7**
NN		**8.7**	—				.2
NNP	.2	**3.3**	**4.1**	—	.2		
RB	**2.2**	2.0	.5		—		
VBD		.3	.5			—	**4.4**
VBN		**2.8**				**2.6**	—

- Create a **confusion matrix**
- See which errors are most common and fix those.
  - Noun (NN) vs ProperNoun (NNP) vs Adj (JJ)
  - Preterite (VBD) vs Participle (VBN) vs Adjective (JJ)

Evaluation

- The result is compared with a manually coded “Gold Standard”
- Typically accuracy reaches 96-97%
- This may be compared with result for a baseline tagger (one that uses no context or less context).
- Important: 100% is impossible even for human annotators!

Brill Taggers and Transformation Based Learning (TBL)

Rule-based vs. HMM-based taggers

1. Rule-based taggers are nice because they capture a lot of explicit linguistic knowledge in a way that makes sense (and might be useable by other tools later in the pipeline).
2. But they're expensive and slow to build and require a lot of human effort.
3. HMM-based taggers, by contrast, require only a tagged corpus and are fast to train.

Potential problems with HMM-based taggers

Brill (1995) identified two problems with HMM-based taggers:

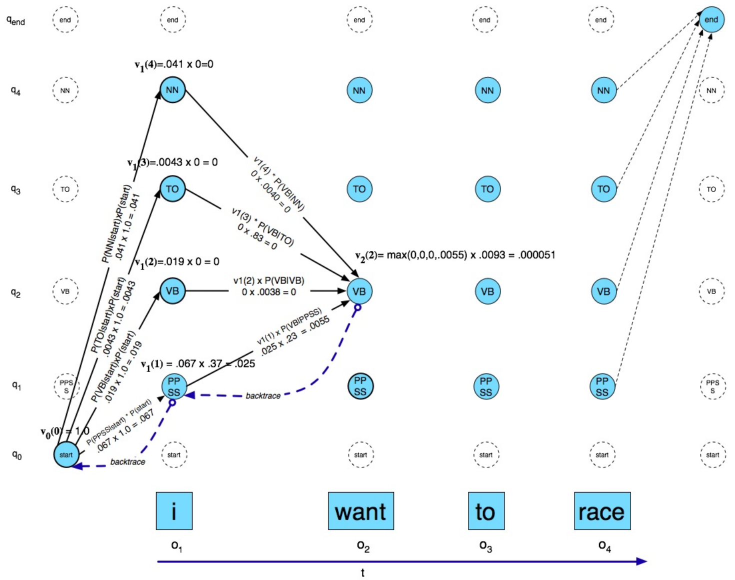
1. "stochastic taggers have the disadvantage that linguistic information is captured only indirectly, in large tables of statistics."
2. The relationships modelled by an HMM are between **tags** in a sequence and not between **words** in a sequence.

Potential problems with HMM-based taggers

In HMM taggers,

- "state transition probabilities ($P(T_i|T_{i-1} \ldots T_{i-n})$) express the likelihood of a tag immediately following n other tags,
- and emit probabilities ($P(W_j|T_i)$) express the likelihood of a word, given a tag.
- Many useful relationships, such as that between a word and the previous word, or between a tag and the following word, are not directly captured by Markov-model based taggers." (Brill 1995, p. 555)

Think about HMM-based taggers work...



Solution: Combine Them!

- Use a simple stochastic (e.g. n-gram) tagger to provide an initial set of tags and then
- Apply a set of transformation rules to correct errors in these tag assignments.
- better still...

Solution: Combine Them!

Don't write the rules by hand! Learn them from a training corpus.

1. Assign most probable tag to each word (e.g. $P(DET|The)$) in a corpus.
2. Compare automatic tag assignments to a hand-tagged version of that same corpus.
3. Learn rules to correct the most frequent errors.
4. Repeat

Machine Learning: three kinds of learning

Supervised Learn from entirely human-tagged data.

Unsupervised Induce rules without recourse to human-tagged data.

Semi-Supervised Bootstrap with human-tagged data, generalize to new data, retrain using both data sets.

Brill (1995) Example transformations learned from Penn Treebank

	Change Tag		
#	From	To	Condition
1	NN	VB	Previous tag is *TO*
2	VBP	VB	One of the previous three tags is *MD*
3	NN	VB	One of the previous two tags is *MD*
4	VB	NN	One of the previous two tags is *DT*
5	VBD	VBN	One of the previous three tags is *VBZ*
6	VBN	VBD	Previous tag is *PRP*
7	VBN	VBD	Previous tag is *NNP*
8	VBD	VBN	Previous tag is *VBD*
9	VBP	VB	Previous tag is *TO*
10	POS	VBZ	Previous tag is *PRP*
11	VB	VBP	Previous tag is *NNS*
12	VBD	VBN	One of previous three tags is *VBP*
13	IN	WDT	One of next two tags is *VB*
14	VBD	VBN	One of previous two tags is *VB*
15	VB	VBP	Previous tag is *PRP*
16	IN	WDT	Next tag is *VBZ*
17	IN	DT	Next tag is *NN*
18	JJ	NNP	Next tag is *NNP*
19	IN	WDT	Next tag is *VBD*
20	JJR	RBR	Next tag is *JJ*

(12) From **IN** to **RB** if the word two positions to the right is **as**.

(16) From **VBP** to **VB** if one of the previous two words is **n't**.[14]

But how can it learn those rules?

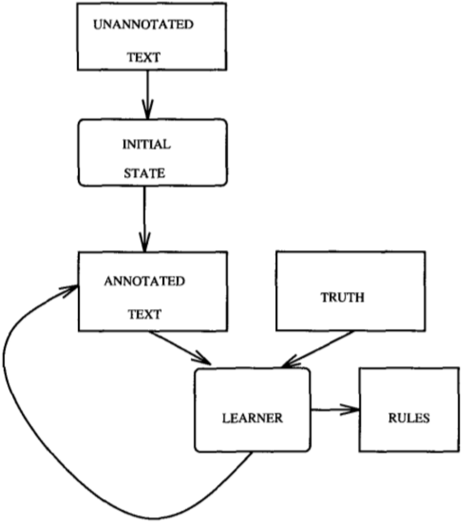
How must this work?
How does the tagger know what kinds of relationships are possible between words and tags?

Templates

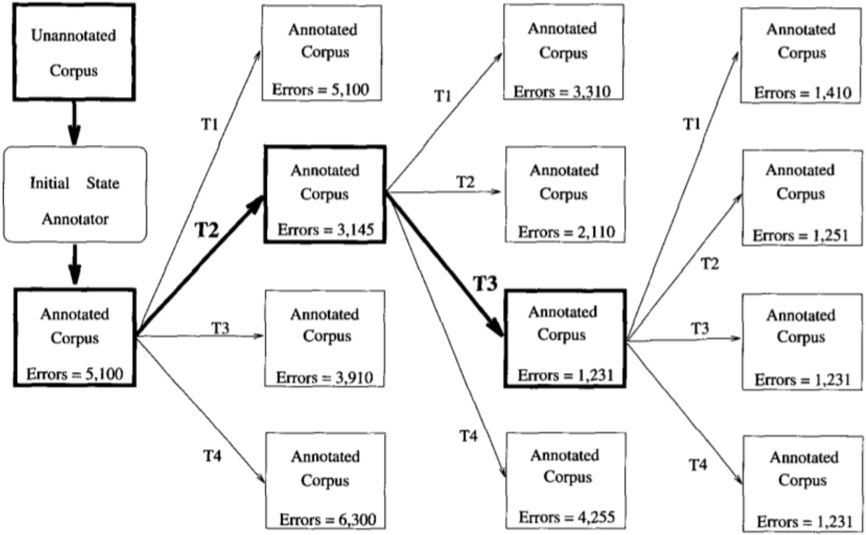
Templates specify the types of linguistic knowledge that can be learned from the data. e.g.: `Change tag` **a** `to tag` **b** `when...`

- `... the (previous|next) tag is` **z**
- `... one of the (previous|next) two tags is` **z**
- `... one of the (previous|next) three tags is` **z**
- `... the (previous|next) word is` **z**
- `... the word three (before|after) is tagged` **z**
- `... the (previous|next) word is tagged` **z** `and the word two (before|after) is tagged` **y**
- etc.

Brill (1995) Figure 1: Error Driven Learning



Brill (1995) Figure 2: Example Error Driven Learning



Brill Demonstration

Brill demonstration using Python and NLTK

For next time:

For next time:

1. Friday: **Hidden Markov Models**
2. **Read:** J&M chapter 6 pp 173 - 183