Now

- POS Tagging
- POS: Part of Speech
- 3 Approaches
  - Rule-based
  - Stochastic-based
  - Transformation-based
- Reading:
  - Jurafsky & Martin Ch. 8: 8.2 → 8.6 incl.

What is Tagging?

- Assigning lexical classes to words
- Why?
- Many words have more than one sense
- Resolving the senses can be helped by knowing the lexical class
  - Book the flight
  - Book the suspect
  - Hand me the book
  - A book of matches
- Brown Corpus
  - Only 11.5% of word types are ambiguous
  - But >40% of word tokens are ambiguous
  - Conclusion?
  - Many common words are ambiguous

- To give an idea of the numbers involved in ambiguity

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	(“still”)

Tagsets

- Penn Treebank Corpus

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

VB	DT	NN	.
Book	that	flight	.

Rule-based Tagging

- ENGTWOL tagger
- 2 stages
  - Use a dictionary to assign words potential parts of speech
  - Then use hand-written disambiguation rules to find a single POS
- Sample lexical entries

Word	POS	Additional POS features
smaller	ADJ	COMPARATIVE
entire	ADJ	ABSOLUTE ATTRIBUTIVE
fast	ADV	SUPERLATIVE
that	DET	CENTRAL DEMONSTRATIVE SG
all	DET	PREDETERMINER SG/PL QUANTIFIER
dog’s	N	GENITIVE SG
furniture	N	NOMINATIVE SG NOINDEFDETERMINER
one-third	NUM	SG
she	PRON	PERSONAL FEMININE NOMINATIVE SG3
show	V	IMPERATIVE VFIN
show	V	PRESENT -SG3 VFIN
show	N	NOMINATIVE SG
shown	PCP2	SVOO SVO SV
occurred	PCP2	SV
occurred	V	PAST VFIN SV

- Write any sentence
  - Get candidate POSs & write rules

Stochastic Tagging

- Pick the most likely tag for a word
- Like N-grams, we use tag sequences
- Use Viterbi algorithm to find most likely tag sequence
- For now we will look at choosing the tag that maximises:
  - $P(\text{word}|\text{tag}) * P(\text{tag}|\text{previous } n \text{ tags})$
  - Note simplification
- Bigram tagger
  - $t_i = \text{argmax}_j P(t_j|t_i, w_i)$
  - $t_i = \text{argmax}_j P(t_j|t_{i-1})P(w_i|t_j)$

Example

- Brown Corpus
- Secretariat/NNP is/BBZ expected/VBN to/TO **race**/VB tomorrow/NN
- People/NNS continue/VBP to/TO inquire/VB the/DT reason/NN for/IN the/DT **race**/NN for/IN outer/JJ space/NN
- Bigram
  - to/TO **race**/??
  - the/DT **race**/??

Bigram Example

- $t_i = \text{argmax}_j P(t_j|t_{i-1})P(w_i|t_j)$
- Which is greater?
- $P(\text{VB}|\text{TO})P(\text{race}|\text{VB})$
- $P(\text{NN}|\text{TO})P(\text{race}|\text{NN})$
- $P(\text{VB}|\text{TO}) = 0.34$
- $P(\text{NN}|\text{TO}) = 0.021$
- $P(\text{race}|\text{VB}) = 0.00003$
- $P(\text{race}|\text{NN}) = 0.00041$
- $P(\text{VB}|\text{TO})P(\text{race}|\text{VB}) = 0.00001$
- $P(\text{NNTO})P(\text{race}|\text{NN}) = 0.000007$

Transformation-based Tagging

- a.k.a. Brill Tagger
- Based on rules
- Rules are learnt automatically – machine learning
- Painting analogy
- Choose most useful rule first – common theme in machine learning
  - e.g. 20 questions

How Rules are Applied

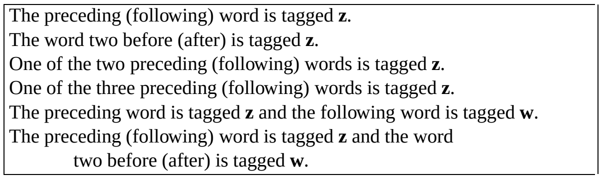
- First label words with most likely tag
  - $P(\text{NN}|\text{race}) = 0.98$
  - $P(\text{VB}|\text{race}) = 0.02$
- i.e. label *race* as NN
  - WRONG: is/BBZ expected/VBN to/TO **race/NN** tomorrow/NN
  - RIGHT: the/DT **race/NN** for/IN outer/JJ space/NN
- Then apply learnt transformation rules
  - Change NN to VB when previous tag is TO
  - expected/VBN to/TO **race/NN** → expected/VBN to/TO **race/VB**

How Rules are Learnt

- 3-step approach
  1. Label words with most likely tag
  2. Examine every possible transformation (rule) & select transformation that gives most improved tagging
  3. Retag data with this rule
  4. Go back to step 2 unless some stopping criterion is met
- Supervised learning
  - Needs correctly tagged data from which to learn
- Output:
  - Ordered list of transformations
  - This is the tagging procedure
- Has produced impressive results

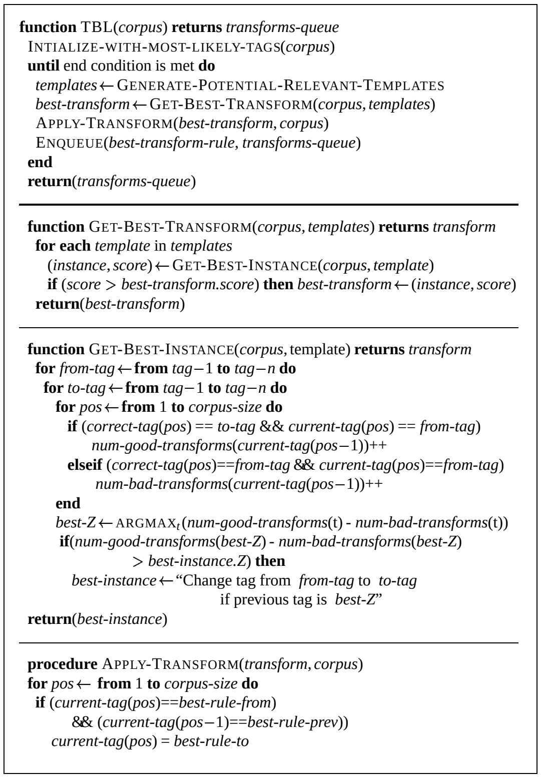
Templates

- Number of transformations is potentially unlimited
- So design a small number of transformation templates
- e.g. Change tag **a** to tag **b** when...



The preceding (following) word is tagged **z**.
The word two before (after) is tagged **z**.
One of the two preceding (following) words is tagged **z**.
One of the three preceding (following) words is tagged **z**.
The preceding word is tagged **z** and the following word is tagged **w**.
The preceding (following) word is tagged **z** and the word
two before (after) is tagged **w**.

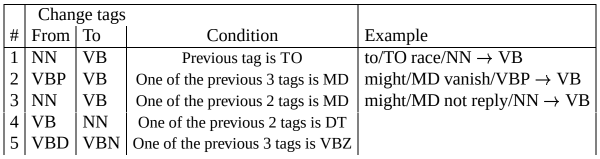
Algorithm



```
function TBL(corpus) returns transforms-queue
  INTIALIZE-WITH-MOST-LIKELY-TAGS(corpus)
  until end condition is met do
    templates ← GENERATE-POTENTIAL-RELEVANT-TEMPLATES
    best-transform ← GET-BEST-TRANSFORM(corpus, templates)
    APPLY-TRANSFORM(best-transform, corpus)
    ENQUEUE(best-transform-rule, transforms-queue)
  end
  return(transforms-queue)

function GET-BEST-TRANSFORM(corpus, templates) returns transform
  for each template in templates
    (instance, score) ← GET-BEST-INSTANCE(corpus, template)
    if (score > best-transform.score) then best-transform ← (instance, score)
  return(best-transform)

function GET-BEST-INSTANCE(corpus, template) returns transform
  for from-tag ← from tag−1 to tag−n do
    for to-tag ← from tag−1 to tag−n do
      for pos ← from 1 to corpus-size do
        if (correct-tag(pos) == to-tag && current-tag(pos) == from-tag)
            num-good-transforms(current-tag(pos−1))++
        elseif (correct-tag(pos)==from-tag && current-tag(pos)==from-tag)
            num-bad-transforms(current-tag(pos−1))++
      end
      best-Z ← ARGMAX_t(num-good-transforms(t) - num-bad-transforms(t))
      if(num-good-transforms(best-Z) - num-bad-transforms(best-Z)
                > best-instance.Z) then
        best-instance ← “Change tag from  from-tag to  to-tag
                          if previous tag is  best-Z”
  return(best-instance)

procedure APPLY-TRANSFORM(transform, corpus)
  for pos ← from 1 to corpus-size do
    if (current-tag(pos)==best-rule-from)
        && (current-tag(pos−1)==best-rule-prev))
      current-tag(pos) = best-rule-to
```

Results

- Some rules learnt by Brill’s original tagger:



	Change tags			
#	From	To	Condition	Example
1	NN	VB	Previous tag is TO	to/TO race/NN → VB
2	VBP	VB	One of the previous 3 tags is MD	might/MD vanish/VBP → VB
3	NN	VB	One of the previous 2 tags is MD	might/MD not reply/NN → VB
4	VB	NN	One of the previous 2 tags is DT	
5	VBD	VBN	One of the previous 3 tags is VBZ	