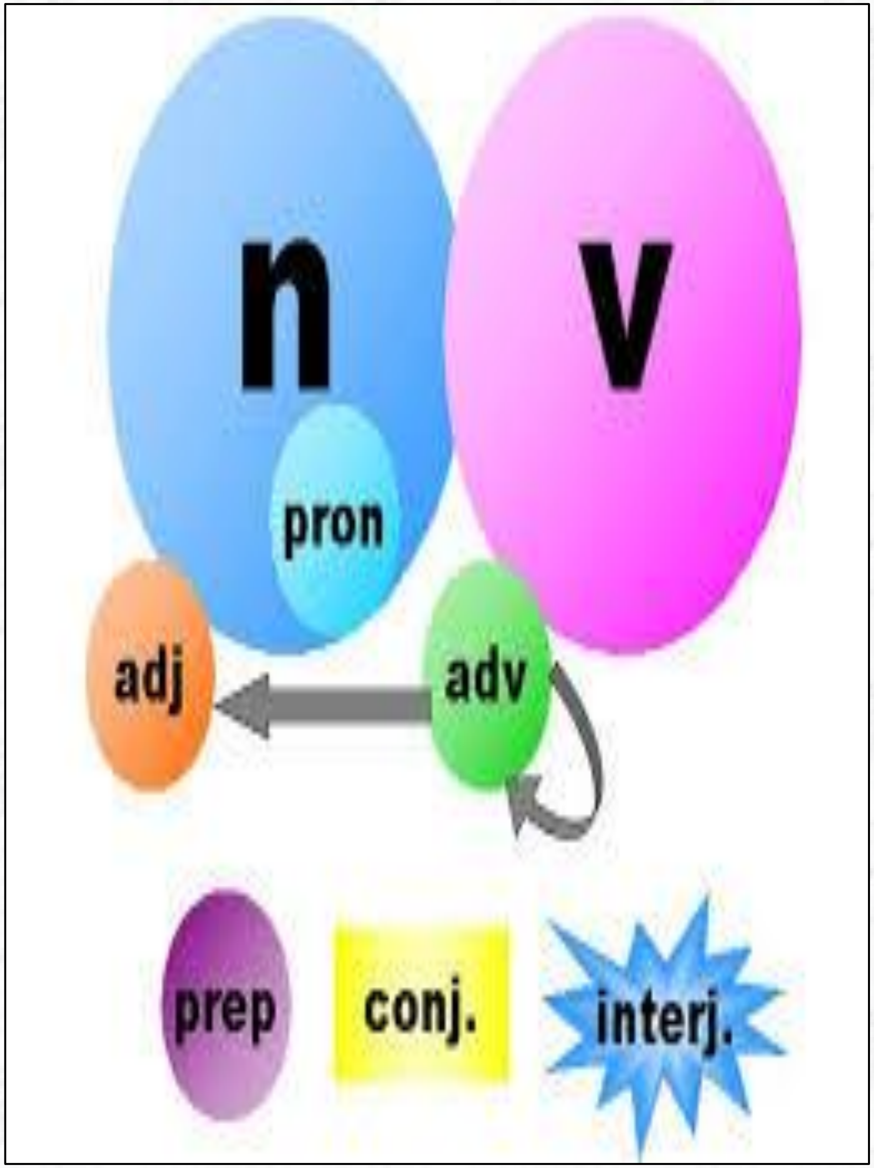
Part of speech tagging



Part Of Speech Tagging Methods

Instructor : Dr. Hanaa Bayomi Ali
Mail : h.mobarz @ fci-cu.edu.eg

POS Tagging

The Tagging Task

Input : `the lead paint is unsafe`

Output: `the/Det lead/N paint/N is/V unsafe/Adj`

POS Tagging Methods:

1. Manual Tagging
2. Machine Tagging
3. A Combination of Both

Manual Tagging

Methods:

1. Agree on a **Tagset** after much discussion.
2. Chose a corpus, annotate it manually by two or more people.
3. Check on inter-annotator agreement.
4. Fix any problems with the Tagset (if still possible).

Machine Tagging

1. Rule based tagging.
2. Stochastic tagging.

1- Rule-based tagging

<u>A Two-stage architecture</u>

1- Use lexicon FST (dictionary) to tag each word
with all possible POS
- Apply hand-written rules to eliminate tags.

2- The rules eliminate tags that are inconsistent with the context, and should reduce the list of POS tags to a single POS per word.

Start with a dictionary

- she: PRP — Personal pronoun
- promised: VBN,VBD — Past-participle , past tens
- To: TO
- back: VB, JJ, RB, NN — Verb,Adjective,adverb,Noun
- the: DT — Determine
- bill: NN, VB — Verb,Noun

- Etc… for the ~100,000 words of English

Use the dictionary to assign every possible tag

			NN		
			RB		
	VBN		JJ		VB
PRP	VBD	TO	VB	DT	NN
She	**promised**	**to**	**back**	**the**	**bill**

Write rules to eliminate tags

Eliminate VBN if VBD is an option when VBN|VBD follows “<start> PRP”

			NN		
			RB		
			JJ		VB
PRP	VBD	TO	VB	DT	NN
She	**promised**	**to**	**back**	**the**	**bill**

The ENGTWOL tagger

- Morphology for lemmatization.
- 56 000 entries for English word stems (first pass)
- 1100 handwritten constraints to eliminate tags (second pass)

Sample ENGTWOL Lexicon

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

2- Stochastic Tagging

- Based on probability of certain tag occurring given various possibilities

- Requires a training corpus

- Simple Method: Choose most frequent tag in training text for each word!

- HMM is an example

The Most Frequent Tag algorithm

- For each word
  - Create dictionary with each possible tag for a word
  - Take a tagged corpus
  - Count the number of times each tag occurs for that word
- Given a new sentence
  - For each word, pick the most frequent tag for that word from the corpus.

Hidden Markov Map (HMM)

Making some simplifying Markov assumptions, the basic HMM equation for a single tag is:

$t_i = \text{argmax}_i \; P(t_i \mid t_{i-1}) * P(w_i \mid t_i)$

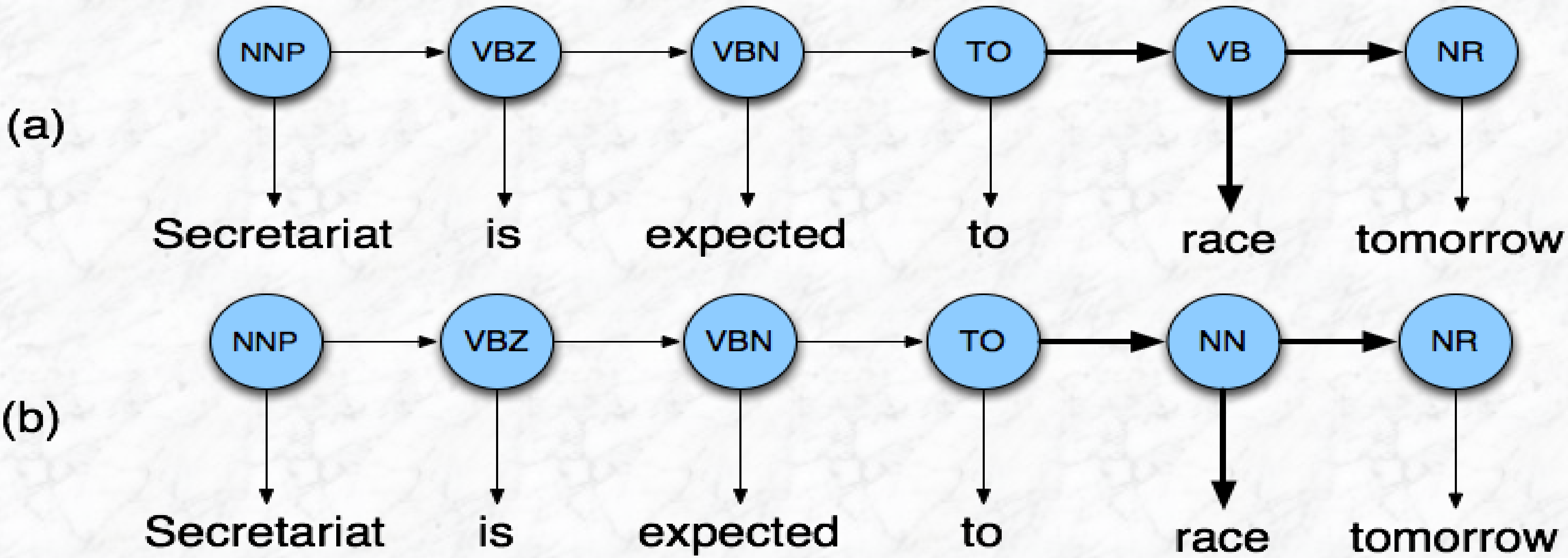
- -The function $\text{argmax}_x F(x)$ means "the x such that F(x) is maximized"
- -The first P is the tag sequence probability, the second is the word likelihood given the tag.

Most of the **better statistical models report around 95% accuracy on standard datasets**

But, note you get **91% accuracy just by picking the most likely tag!**

A Simple Example

From the Brown Corpus



Assume previous words have been tagged, and we want to tag the word *race*.

Bigram tagger

- to/TO *race/*?
- the/DT *race/*?

A Simple Example

Goal: choose between **NN** and **VB** for the sequence *to race*

Plug these into our bigram HMM tagging equation:

P(race | VB) * P(VB | TO)

P(race | NN) * P(NN | TO)

How do we compute the tag sequence probabilities and the word likelihoods?

Word Likelihood

We must compute the likelihood of the word race given each tag. I.e., **P(race | VB)** and **P(race | NN)**

Note: we are **NOT** asking which is the most likely tag for the word.

Instead, we are asking, **if we were expecting a verb, how likely is it that this verb would be *race*?**

From the Brown and Switchboard Corpora:

P(*race* | VB) = .00003

P(*race* | NN) = .00041

Tag Sequence Probabilities

Computed from the corpus by counting and normalizing.

We expect VB more likely to follow TO because infinitives (*to race, to eat*) are common in English, but it is possible for NN to follow TO (*walk to school, related to fishing*).

From the Brown and Switchboard corpora:

$P(VB \mid TO) = .340$

$P(NN \mid TO) = .021$

And the Winner is…

Multiplying tag sequence probabilities by word likelihoods gives

P(*race* | VB) = .00003
P(*race* | NN) = .00041
P(VB | TO) = .340
P(NN | TO) = .021

P(*race* | VB) * P(VB | TO) = .000010

P(*race* | NN) * P(NN | TO) = .000007

So, even a simple bigram version correctly tags race as a VB, despite the fact that it is the less likely sense.

And the Winner is…

Multiplying tag sequence probabilities by word likelihoods gives

P(*race* | VB) = .00003
P(*race* | NN) = .00041
P(VB | TO) = .340
P(NN | TO) = .021

P(*race* | VB) * P(VB | TO) = .000010

P(*race* | NN) * P(NN | TO) = .000007

So, even a simple bigram version correctly tags race as a VB, despite the fact that it is the less likely sense.

Statistical POS Tagging (whole sequence)

Challenges

1- Multivariable output

- make multiple prediction simultaneously

2- Variable length output

- Sentence length not fixed

Statistical POS Tagging (whole sequence)

Goal: choose the best sequence of tags T for a sequence of words W in a sentence

$$T' = \arg\max_{T \in \tau} P(T \mid W)$$

By Bayes Rule (giving us something easier to calculate)

$$P(T \mid W) = \frac{P(T)P(W \mid T)}{P(W)}$$

Since we can ignore P(W), we have

$$T' = \arg\max_{T \in \tau} P(T)P(W \mid T)$$

Statistical POS Tagging (whole sequence)

Goal: choose the best sequence of tags T for a sequence of words W in a sentence

$$T' = \underset{T \in \tau}{\arg\max}\, P(T|W)$$

By Bayes Rule (giving us something easier to calculate)

$$P(T|W) = \frac{P(T)P(W|T)}{P(W)}$$

Since we can ignore P(W), we have

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

Statistical POS Tagging: the Prior

$P(T) = P(t_1, t_2, \ldots, t_{n-1}, t_n)$

By the Chain Rule:

$= P(t_n \mid t_1, \ldots, t_{n-1}) \, P(t_1, \ldots, t_{n-1})$

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

Making the Markov assumption:

$\approx P(t_i \mid t_{i-N+1}^{i-1})$ e.g., for bigrams, $\prod_{i=1}^{n} P(t_i \mid t_{i-1})$

Statistical POS Tagging: the (Lexical) Likelihood

$P(W|T) = P(w_1, w_2, \ldots, w_n \mid t_1, t_2, \ldots, t_n)$

From the Chain Rule:

$$= \prod_{i=1}^{n} P(w_i | w_1 t_1 \ldots w_{i-1} t_{i-1} t_i)$$

Simplifying assumption: probability of a word depends only on its own tag $P(w_i|t_i)$

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

So...

$$T' = \underset{T \in \tau}{\arg\max} \prod_{i=1}^{n} P(t_i | t_{i-1}) \prod_{i=1}^{n} P(w_i | t_i)$$

Estimate the Tag Priors and the Lexical Likelihoods from Corpus

Maximum-Likelihood Estimation

For bigrams:

$P(t_i | t_{i-1}) = c(t_{i-1}, t_i) / c(t_{i-1})$

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

Statistical POS Tagging (whole sequence)

- Want to compute
  - $P(T)\,P(W|T) \approx P(t_1)\,P(t_2|t_1) \ldots P(t_n|t_{n-1})\;\; P(w_1|t_1)\,P(w_2|t_2) \ldots P(w_n|t_n)$
- Let
  - $c(t_i)$ = frequency of t_i in the corpus
  - $c(w_i,t_i)$ = frequency of w_i/t_i in the corpus
  - $c(t_{i-1},t_i)$ = frequency of $t_{i-1}\ t_i$ in the corpus
- Then we can use
  - $P(t_i|t_{i-1}) = c(t_{i-1},t_i)/c(t_{i-1})$,
  - $P(w_i|t_i) = c(w_i,t_i)/c(t_i)$

Question

What is the tagging of the following sentence?

Computers process programs accurately

with the following HMM tagger: (part of) lexicon:

computers	N	0.123
process	N	0.1
process	V	0.2
programs	N	0.11
programs	V	0.15
accurately	Adv	0.789

(part of) transitions:

$P(N|V)=0.5$ $P(N|Adv)=0.12$ $P(V|Adv)=0.05$ $P(V|N)=0.4$

$P(Adv|N)=0.01$ $P(Adv|V)=0.13$ $P(N|N)=0.6$ $P(V|V)=0.05$

Answer

Solutions

4 choices (it's a lattice):

```
computers process programs accurately
    N        N        N          Adv
             V        V
```

Differences are (skept the common factors):

```
P(N|N) P(process|N) P(N|N) P(programs|N) P(Adv|N)
P(N|N) P(process|N) P(V|N) P(programs|V) P(Adv|V)
P(V|N) P(process|V) P(N|V) P(programs|N) P(Adv|N)
P(V|N) P(process|V) P(V|V) P(programs|V) P(Adv|V)
```

i.e.:

```
    0.6    0.1    0.6    0.11   0.01
--> 0.6    0.1    0.4    0.15   0.13 <--MAX
    0.4    0.2    0.5    0.11   0.01
    0.4    0.2    0.05   0.15   0.13
```

Tagging obtained (not corresponding to the one expected by an average English reader ;-)):

```
computers process programs accurately
   N        N        V        Adv
```

What is the tagging of the following sentence?

Computers process programs accurately

with the following HMM tagger: (part of) lexicon:

computers	N	0.123
process	N	0.1
process	V	0.2
programs	N	0.11
programs	V	0.15
accurately	Adv	0.789

(part of) transitions:

P(N|V)=0.5	P(N|Adv)=0.12	P(V|Adv)=0.05	P(V|N)=0.4
P(Adv|N)=0.01	P(Adv|V)=0.13	P(N|N)=0.6	P(V|V)=0.05