Part of Speech Tagging & Hidden Markov Models (Part 1)

Mitch Marcus

CSE 391

University of Pennsylvania

NLP Task I – Determining Part of Speech Tags

- **Given a text, assign each token its correct *part of speech (POS) tag*, given its context and a list of *possible* POS tags for each word type**

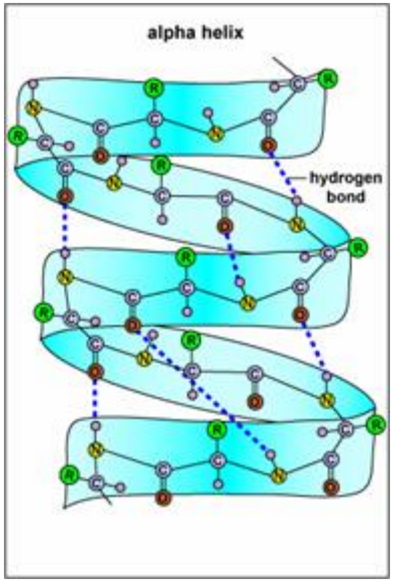
Word	**POS listing in Brown Corpus**		
heat	noun	***verb***	
oil	***noun***		
in	***prep***	noun	adv
a	***det***	noun	noun-proper
large	***adj***	noun	adv
pot	***noun***		

What is POS tagging good for?

- **Speech synthesis:**
  - How to pronounce "lead"?
  - INsult		inSULT
  - OBject		obJECT
  - OVERflow		overFLOW
  - DIScount		disCOUNT
  - CONtent		content
- **Machine Translation**
  - translations of nouns and verbs are different
- **Stemming for search**
  - Knowing a word is a N tells you it gets plurals
  - Can search for "aardvarks" get "aardvark"
- **Parsing and speech recognition and etc**
  - Possessive pronouns (my, your, her) followed by nouns
  - Personal pronouns (I, you, he) likely to be followed by verbs

Equivalent Problem in Bioinformatics

- **Durbin et al. Biological Sequence Analysis, Cambridge University Press.**
- **Several applications, e.g. proteins**
- **From a sequence of amino acids (primary structure): ATCPLELLLD**
- **Infer secondary structure (features of the 3D structure, like helices, sheets, etc.): HHHBBBBBC..**



Penn Treebank Tagset I

Tag	Description	Example
CC	coordinating conjunction	and
CD	cardinal number	1, third
DT	determiner	the
EX	existential *there*	*there* is
FW	foreign word	d'hoevre
IN	preposition/subordinating conjunction	in, of, like
JJ	adjective	green
JJR	adjective, comparative	greener
JJS	adjective, superlative	greenest
LS	list marker	1)
MD	modal	could, will
NN	noun, singular or mass	table
NNS	noun plural	tables (supports)
NNP	proper noun, singular	John
NNPS	proper noun, plural	Vikings

Penn Treebank Tagset II

Tag	Description	Example
PDT	predeterminer	*both* the boys
POS	possessive ending	friend *'s*
PRP	personal pronoun	I, me, him, he, it
PRP$	possessive pronoun	my, his
RB	adverb	however, usually, here, good
RBR	adverb, comparative	better
RBS	adverb, superlative	best
RP	particle	give *up*
TO	*to*	*to* go, *to* him
UH	interjection	uhhuhhuhh

Penn Treebank Tagset III

Tag	Description	Example
VB	verb, base form	take (support)
VBD	verb, past tense	took
VBG	verb, gerund/present participle	taking
VBN	verb, past participle	taken
VBP	verb, sing. present, non-3d	take
VBZ	verb, 3rd person sing. present	takes (supports)
WDT	wh-determiner	which
WP	wh-pronoun	who, what
WP$	possessive wh-pronoun	whose
WRB	wh-abverb	where, when

NLP Task I – Determining Part of Speech Tags

- **The Old Solution:** ***Depth First search.***
  - If each of *n* word tokens has *k* tags on average, try the k^n combinations until one works.

- **Machine Learning Solutions:** ***Automatically learn*** **Part of Speech (POS) assignment.**
  - The best techniques achieve 97+% accuracy per word on new materials, given a POS-tagged training corpus of 10^6 tokens and a set of ~40 POS tags

Simple Statistical Approaches: Idea 1

Simply assign each word its most likely POS.

Success rate: 91%!

Word	**POS listings in Brown**		
heat	noun/89	**verb/5**	
oil	**noun/87**		
in	**prep/20731**	noun/1	adv/462
a	**det/22943**	noun/50	noun-proper/30
large	**adj/354**	noun/2	adv/5
pot	**noun/27**		

Simple Statistical Approaches: Idea 2

For a string of words

$$W = w_1 w_2 w_3 \ldots w_n$$

find the string of POS tags

$$T = t_1\, t_2\, t_3 \ldots t_n$$

which maximizes $P(T \mid W)$

- i.e., the most likely POS tag t_i for each word w_i given its surrounding context

The Sparse Data Problem …

A Simple, Impossible Approach to Compute $P(T|W)$:

Count up instances of the string "heat oil in a large pot" in the training corpus, and pick the *most common tag assignment* to the string..

A BOTEC Estimate of What Works

What parameters can we estimate with a million words of hand tagged training data?

- Assume a uniform distribution of 5000 words and 40 part of speech tags..

Event	Count	Estimate Quality?
tags	40	Excellent
tag bigrams	1600	Excellent
tag trigrams	64,000	OK
tag 4-grams	2.5M	Poor
words	5000	Very Good
word bigrams	25M	Poor
word x tag pairs	200,000	OK

We can get reasonable estimates of

- ***Tag*** **bigrams**
- ***Word x tag pairs***

Bayes Rule plus Markov Assumptions yields a practical POS tagger!

I. By Bayes Rule

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

II. So we want to find

$$\arg\max_T P(T \mid W) = \arg\max_T P(W \mid T) * P(T)$$

III. To compute *P(W|T)*:

- use the chain rule + a Markov assumption
- Estimation requires word x tag and tag counts

IV. To compute *P(T)*:

- use the chain rule + a slightly different Markov assumption
- Estimation requires tag unigram and bigram counts

IV. To compute *P(T)*:

I. **By the chain rule,**

$$P(T) = P(t_1) * P(t_2 \mid t_1) * P(t_3 \mid t_1 t_2) * \ldots * P(t_n \mid t_1 \ldots t_{n-1})$$

II. **Applying the 1st order Markov Assumption**

$$P(T) = P(t_1) * P(t_2 \mid t_1) * P(t_3 \mid t_2) * \ldots * P(t_n \mid t_{n-1})$$

Estimated using tag bigrams/tag unigrams!

III. To compute *P(W|T)*:

I. Assume that the words w_i are conditionally independent given the tag sequence $T = t_1 t_2 \ldots t_n$: $$P(W \mid T) = \prod_{i=1}^{n} P(w_i \mid T)$$

II. Applying a zeroth-order Markov Assumption:

$$P(w_i \mid T) = P(w_i \mid t_i)$$

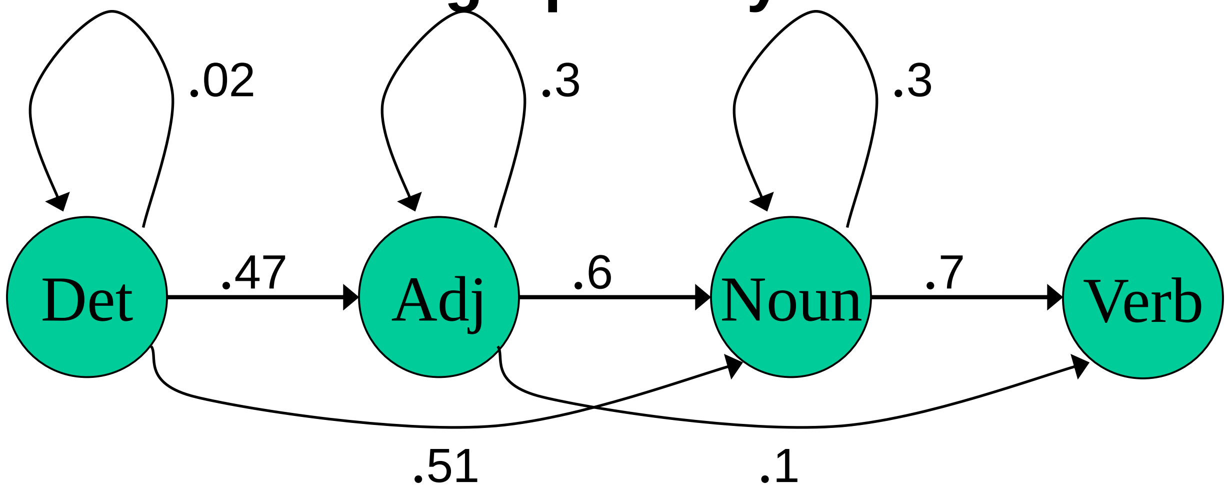
by which $$P(W \mid T) = \prod_{i=1}^{n} P(w_i \mid t_i)$$

So, for a given string $W = w_1 w_2 w_3 \ldots w_n$, the tagger needs to ***find the string of tags T which maximizes***

$$P(T) * P(W|T) \;=$$
$$P(t_1) * P(t_2|t_1) * P(t_3|t_2) * \ldots * P(t_n|t_{n-1}) *$$
$$P(w_1|t_1) * P(w_2|t_2) * \ldots * P(w_n|t_n)$$

Hidden Markov Models

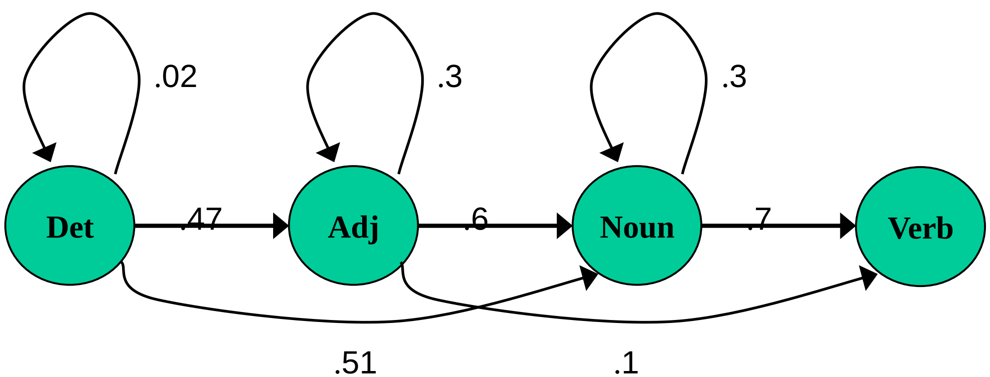
This model is an instance of a Hidden Markov Model. Viewed graphically:



P(w|Det)		P(w|Adj)		P(w|Noun)	
a	.4	good	.02	price	.001
the	.4	low	.04	deal	.0001

Viewed as a generator, an HMM:

- Starts in some initial state t_1 with probability $\pi(t_1)$,
- On each move goes from state t_i to state t_j according to transition probability $a(t_i, t_j)$.
- At each state t_i, it emits a symbol w_k according to the emit probabilities $b(t_i, w_k)$.



P(w|Det)	
a	.4
the	.4

P(w|Adj)	
good	.02
low	.04

P(w|Noun)	
price	.001
deal	.0001

Recognition using an HMM

I. **By Bayes Rule**

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

II. **We select the Tag sequence T that maximizes** *P(T|W):*

$$\arg\max_{T} P(T\,|\,W)$$

$$= \arg \max_{T=t_1t_2\ldots t_n} P(T) * P(W\,|\,T)$$

$$= \arg \max_{T=t_1t_2\ldots t_n} \pi(t_1) * \prod_{i=1}^{n-1} a(t_i, t_{i+1}) * \prod_{i=1}^{n} b(t_i, w_i)$$

Training and Performance

- **To estimate the parameters of this model, given an annotated training corpus use the MLE:**

To estimate $\mathrm{P}(\mathrm{t_i}|\mathrm{t_{i-1}})$:

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

To estimate $\mathrm{P}(\mathrm{w_i}|\mathrm{t_i})$:

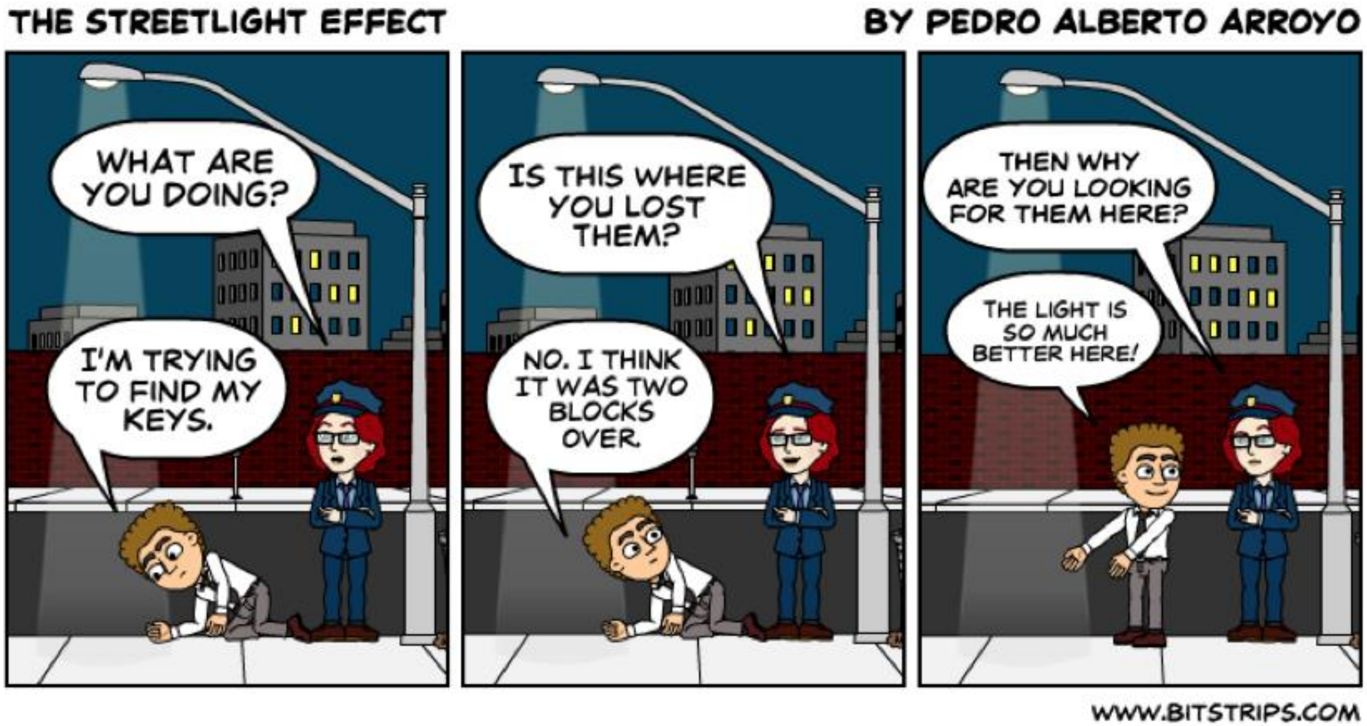
$$\frac{Count(\ w_i \text{ tagged } t_i)}{Count(\text{ all words tagged } t_i)}$$

- **Because many of these counts are small, *smoothing* is necessary for best results…**
- **Such taggers typically achieve about 95-96% correct tagging,** for the standard 40-tag POS set.

POS from bigram and word-tag pairs??

A Practical compromise

- **Rich Models often require vast amounts of data**
- **Well estimated bad models often outperform badly estimated better models**



Practical Tagging using HMMs

- **Finding this maximum can be done using an exponential search through all strings for T.**

- **However, there is a *linear time* solution using *dynamic programming* called *Viterbi decoding*.**