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

Mitch Marcus
CSE 391
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

<table>
<thead>
<tr>
<th>Word</th>
<th>POS listing in Brown Corpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>heat</td>
<td>noun, <em>verb</em></td>
</tr>
<tr>
<td>oil</td>
<td><em>noun</em></td>
</tr>
<tr>
<td>in</td>
<td><em>prep</em>, noun, adv</td>
</tr>
<tr>
<td>a</td>
<td><em>det</em>, noun, noun-proper</td>
</tr>
<tr>
<td>large</td>
<td><em>adj</em>, noun, adv</td>
</tr>
<tr>
<td>pot</td>
<td><em>noun</em></td>
</tr>
</tbody>
</table>
What is POS tagging good for?

- **Speech synthesis:**
  - How to pronounce “lead”?  
  - INsult inSULT
  - OObjective 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

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

CIS 391 - Intro to AI
# Penn Treebank Tagset II

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDT</td>
<td>predeterminer</td>
<td><em>both</em> the boys</td>
</tr>
<tr>
<td>POS</td>
<td>possessive ending</td>
<td><em>friend 's</em></td>
</tr>
<tr>
<td>PRP</td>
<td>personal pronoun</td>
<td><em>I, me, him, he, it</em></td>
</tr>
<tr>
<td>PRP$</td>
<td>possessive pronoun</td>
<td><em>my, his</em></td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
<td><em>however, usually, here, good</em></td>
</tr>
<tr>
<td>RBR</td>
<td>adverb, comparative</td>
<td><em>better</em></td>
</tr>
<tr>
<td>RBS</td>
<td>adverb, superlative</td>
<td><em>best</em></td>
</tr>
<tr>
<td>RP</td>
<td>particle</td>
<td><em>give up</em></td>
</tr>
<tr>
<td>TO</td>
<td><em>to</em></td>
<td><em>to go, to him</em></td>
</tr>
<tr>
<td>UH</td>
<td>interjection</td>
<td><em>uhhuhhhhh</em></td>
</tr>
</tbody>
</table>
## Penn Treebank Tagset III

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

<table>
<thead>
<tr>
<th>Word</th>
<th>POS listings in Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td>heat</td>
<td>noun/89, verb/5</td>
</tr>
<tr>
<td>oil</td>
<td>noun/87</td>
</tr>
<tr>
<td>in</td>
<td>prep/20731, noun/1, adv/462</td>
</tr>
<tr>
<td>a</td>
<td>det/22943, noun/50, noun-proper/30</td>
</tr>
<tr>
<td>large</td>
<td>adj/354, noun/2, adv/5</td>
</tr>
<tr>
<td>pot</td>
<td>noun/27</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Event</th>
<th>Count</th>
<th>Estimate</th>
<th>Quality?</th>
</tr>
</thead>
<tbody>
<tr>
<td>tags</td>
<td>40</td>
<td></td>
<td>Excellent</td>
</tr>
<tr>
<td>tag bigrams</td>
<td>1600</td>
<td></td>
<td>Excellent</td>
</tr>
<tr>
<td>tag trigrams</td>
<td>64,000</td>
<td></td>
<td>OK</td>
</tr>
<tr>
<td>tag 4-grams</td>
<td>2.5M</td>
<td></td>
<td>Poor</td>
</tr>
<tr>
<td>words</td>
<td>5000</td>
<td></td>
<td>Very Good</td>
</tr>
<tr>
<td>word bigrams</td>
<td>25M</td>
<td></td>
<td>Poor</td>
</tr>
<tr>
<td>word x tag pairs</td>
<td>200,000</td>
<td></td>
<td>OK</td>
</tr>
</tbody>
</table>

We can get reasonable estimates of
- Tag bigrams
- Word x tag pairs

CIS 391 - Intro to AI
Bayes Rule plus Markov Assumptions yields a practical POS tagger!

I. By Bayes Rule

\[ P(T \mid W) = \frac{P(W \mid T) \times P(T)}{P(W)} \]

II. So we want to find

\[ \arg \max_T P(T \mid W) = \arg \max_T P(W \mid T) \times P(T) \]

III. To compute \( P(W \mid T) \):

- 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,

$$P(T) = P(t_1) \times P(t_2 \mid t_1) \times P(t_3 \mid t_1t_2) \times \ldots \times P(t_n \mid t_1...t_{n-1})$$

II. Applying the 1st order Markov Assumption

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

*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:
\[ P(w_i \mid T) = P(w_i \mid t_i) \]
\[ \text{by which } P(W \mid T) = \prod_{i=1}^{n} P(w_i \mid t_i) \]

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 \[ P(T) \ast P(W \mid T) = P(t_1) \ast P(t_2 \mid t_1) \ast P(t_3 \mid t_2) \ast \ldots \ast P(t_n \mid t_{n-1}) \ast P(w_1 \mid t_1) \ast P(w_2 \mid t_2) \ast \ldots \ast P(w_n \mid t_n) \]
Hidden Markov Models

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

\[
\begin{align*}
\text{Det} & \xrightarrow{.47} \text{Adj} \\
\text{Adj} & \xrightarrow{.6} \text{Noun} \\
\text{Noun} & \xrightarrow{.7} \text{Verb} \\
\end{align*}
\]

\[
\begin{align*}
P(w|\text{Det}) & \\
\text{a} & .4 \\
\text{the} & .4 \\
\end{align*}
\]

\[
\begin{align*}
P(w|\text{Adj}) & \\
\text{good} & .02 \\
\text{low} & .04 \\
\end{align*}
\]

\[
\begin{align*}
P(w|\text{Noun}) & \\
\text{price} & .001 \\
\text{deal} & .0001 \\
\end{align*}
\]
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 $b(t_i, w_k)$.
Recognition using an HMM

I. By Bayes Rule

\[ P(T \mid W) = \frac{P(T) \cdot P(W \mid T)}{P(W)} \]

II. We select the Tag sequence T that maximizes \( P(T \mid W) \):

\[
\arg\max_T P(T \mid W) = \arg\max_{T=t_1t_2\ldots t_n} P(T) \cdot P(W \mid T) \\
= \arg\max_{T=t_1t_2\ldots t_n} \pi(t_1) \cdot \prod_{i=1}^{n-1} a(t_i, t_{i+1}) \cdot \prod_{i=1}^{n} b(t_i, w_i)
\]
Training and Performance

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

\[
\text{To estimate } P(t_i|t_{i-1}):\ \\
\frac{\text{Count}(t_{i-1}t_i)}{\text{Count}(t_{i-1})}
\]

\[
\text{To estimate } P(w_i|t_i): \\
\frac{\text{Count}(w_i \text{ tagged } t_i)}{\text{Count}(\text{ all words tagged } t_i)}
\]

- 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

THE STREETLIGHT EFFECT

BY PEDRO ALBERTO ARROYO

WHAT ARE YOU DOING?
I'M TRYING TO FIND MY KEYS.

IS THIS WHERE YOU LOST THEM?
NO. I THINK IT WAS TWO BLOCKS OVER.

THEN WHY ARE YOU LOOKING FOR THEM HERE?
THE LIGHT IS SO MUCH BETTER HERE!

WWW.BITSTRIPS.COM
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*.