Speech and Language Processing

SLP Chapter 5

Part-of-Speech Tagging
A simple but useful form of linguistic analysis

(based on the slides of Jurafsky & Martin and Manning)

Today

- Parts of speech (POS)
- Tagsets
- POS Tagging
  - Rule-based tagging
  - Probabilistic (HMM) tagging
  - Hybrid tagging (optional)
Parts of Speech

- 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.

POS 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
POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

```
the koala put the keys on the table
```

```
the DET koala put the keys on the table
```
POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

```plaintext
the DET
koala N
put V
the DET
keys N
on V
the DET
table N
```
POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

<table>
<thead>
<tr>
<th>WORD</th>
<th>tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>koala</td>
<td>N</td>
</tr>
<tr>
<td>put</td>
<td>V</td>
</tr>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>keys</td>
<td>N</td>
</tr>
<tr>
<td>on</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td></td>
</tr>
<tr>
<td>table</td>
<td></td>
</tr>
</tbody>
</table>
## POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

<table>
<thead>
<tr>
<th>WORD</th>
<th>tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>koala</td>
<td>N</td>
</tr>
<tr>
<td>put</td>
<td>V</td>
</tr>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>keys</td>
<td>N</td>
</tr>
<tr>
<td>on</td>
<td>P</td>
</tr>
<tr>
<td>the</td>
<td></td>
</tr>
<tr>
<td>table</td>
<td></td>
</tr>
</tbody>
</table>
POS Tagging

- The process of assigning a part-of-speech or lexical class marker to each word in a collection.

<table>
<thead>
<tr>
<th>WORD</th>
<th>tag</th>
</tr>
</thead>
<tbody>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>koala</td>
<td>N</td>
</tr>
<tr>
<td>put</td>
<td>V</td>
</tr>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>keys</td>
<td>N</td>
</tr>
<tr>
<td>on</td>
<td>P</td>
</tr>
<tr>
<td>the</td>
<td>DET</td>
</tr>
<tr>
<td>table</td>
<td>N</td>
</tr>
</tbody>
</table>

Why is POS Tagging Useful?

- First step of many practical tasks, e.g.
- Speech synthesis (aka text to speech)
  - How to pronounce "lead"?
  - Object object
  - Content content

- Parsing
  - Need to know if a word is an N or V before you can parse

- Information extraction
  - Finding names, relations, etc.

- Language modeling
  - Backoff
Why is POS Tagging Difficult?

- Words often have more than one POS: 
  - *back*
    - The *back* door = adjective
    - On my *back* =
    - Win the voters *back* =
    - Promised to *back* the bill =
Why is POS Tagging Difficult?

- Words often have more than one POS:
  - *back*
    - The *back* door = adjective
    - On my *back* = noun
    - Win the voters *back* = adverb
    - Promised to *back* the bill = verb

- The POS tagging problem is to determine the POS tag for a particular instance of a word.
POS Tagging

- Input: Plays well with others
- Ambiguity: NNS/VBZ UH/JJ/NN/RB IN NNS
- Output: Plays/VBZ well/RB with/IN others/NNS

---

POS tagging performance

- How many tags are correct? (Tag accuracy)
  - About 97% currently
  - But baseline is already 90%
    - Baseline is performance of stupidest possible method
    - Tag every word with its most frequent tag
    - Tag unknown words as nouns
  - Partly easy because
    - Many words are unambiguous
    - You get points for them (the, a, etc.) and for punctuation marks!
Deciding on the correct part of speech can be difficult even for people

- Mrs/NNP Shaefer/NNP never/RB got/VBD around/RP to/TO joining/VBG

- All/Dt we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DT corner/NN

- Chateau/NNP Petrus/NNP costs/VBZ around/RB 250/CD

How difficult is POS tagging?

- About 11% of the word types in the Brown corpus are ambiguous with regard to part of speech
- But they tend to be very common words. E.g., that
  - I know that he is honest = IN
  - Yes, that play was nice = DT
  - You can’t go that far = RB
- 40% of the word tokens are ambiguous
Open vs. Closed Classes

- **Closed class:** *why?*
  - Determiners: a, an, the
  - Prepositions: of, in, by, ...
  - Auxiliaries: may, can, will had, been, ...
  - Pronouns: I, you, she, mine, his, them, ...
  - Usually **function words** (short common words which play a role in grammar)

- **Open class:** *why?*
  - English has 4: Nouns, Verbs, Adjectives, Adverbs
  - Many languages have these 4, but not all!

Open vs. Closed Classes

- **Closed class:** a small fixed membership
  - Determiners: a, an, the
  - Prepositions: of, in, by, ...
  - Auxiliaries: may, can, will had, been, ...
  - Pronouns: I, you, she, mine, his, them, ...
  - Usually **function words** (short common words which play a role in grammar)

- **Open class:** new ones can be created all the time
  - English has 4: Nouns, Verbs, Adjectives, Adverbs
  - Many languages have these 4, but not all!
**Open Class Words**

- **Nouns**
  - Proper nouns (Pittsburgh, Patti Beeson)
    - English capitalizes these.
  - Common nouns (the rest).
  - Count nouns and mass nouns
    - Count: have plurals, get counted: goat/goats, one goat, two goats
    - Mass: don’t get counted (snow, salt, communism) (*two snows)

- **Adverbs: tend to modify things**
  - Unfortunately, John walked home extremely slowly yesterday
  - Directional/locative adverbs (here, home, downhill)
  - Degree adverbs (extremely, very, somewhat)
  - Manner adverbs (slowly, slinkily, delicately)

- **Verbs**
  - In English, have morphological affixes (eat/eats/eaten)
Closed Class Words

Examples:
- prepositions: on, under, over, ...
- particles: up, down, on, off, ...
- determiners: a, an, the, ...
- pronouns: she, who, I, ...
- conjunctions: and, but, or, ...
- auxiliary verbs: can, may should, ...
- numerals: one, two, three, third, ...

Prepositions from CELEX

<table>
<thead>
<tr>
<th>of</th>
<th>540,085</th>
<th>through</th>
<th>14,964</th>
<th>worth</th>
<th>1,563</th>
<th>pace</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>in</td>
<td>331,235</td>
<td>after</td>
<td>13,670</td>
<td>toward</td>
<td>1,390</td>
<td>nigh</td>
<td>9</td>
</tr>
<tr>
<td>for</td>
<td>142,421</td>
<td>between</td>
<td>13,275</td>
<td>plus</td>
<td>750</td>
<td>re</td>
<td>4</td>
</tr>
<tr>
<td>to</td>
<td>125,691</td>
<td>under</td>
<td>9,525</td>
<td>till</td>
<td>686</td>
<td>mid</td>
<td>3</td>
</tr>
<tr>
<td>with</td>
<td>124,965</td>
<td>per</td>
<td>6,515</td>
<td>amongst</td>
<td>525</td>
<td>o'er</td>
<td>2</td>
</tr>
<tr>
<td>on</td>
<td>109,129</td>
<td>among</td>
<td>5,090</td>
<td>via</td>
<td>351</td>
<td>but</td>
<td>0</td>
</tr>
<tr>
<td>at</td>
<td>100,169</td>
<td>within</td>
<td>5,030</td>
<td>amid</td>
<td>222</td>
<td>ere</td>
<td>0</td>
</tr>
<tr>
<td>by</td>
<td>77,794</td>
<td>towards</td>
<td>4,700</td>
<td>underneath</td>
<td>164</td>
<td>less</td>
<td>0</td>
</tr>
<tr>
<td>from</td>
<td>74,843</td>
<td>above</td>
<td>3,056</td>
<td>versus</td>
<td>113</td>
<td>midst</td>
<td>0</td>
</tr>
<tr>
<td>about</td>
<td>38,428</td>
<td>near</td>
<td>2,026</td>
<td>amidst</td>
<td>67</td>
<td>o'</td>
<td>0</td>
</tr>
<tr>
<td>than</td>
<td>20,210</td>
<td>off</td>
<td>1,695</td>
<td>sans</td>
<td>20</td>
<td>thru</td>
<td>0</td>
</tr>
<tr>
<td>over</td>
<td>18,071</td>
<td>past</td>
<td>1,575</td>
<td>circa</td>
<td>14</td>
<td>vice</td>
<td>0</td>
</tr>
</tbody>
</table>
POS Tagging
Choosing a Tagset

- There are so many parts of speech, potential distinctions we can draw
- To do POS tagging, we need to choose a standard set of tags to work with
- Could pick very coarse tagsets
- More commonly used set is finer grained, the "Penn TreeBank tagset", 45 tags
- Even more fine-grained tagsets exist

### Penn TreeBank POS Tagset

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>Example</th>
<th>Tag</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>coord. conjunction</td>
<td>and, but, or</td>
<td>SYM</td>
<td>symbol</td>
<td>+, %, &amp;</td>
</tr>
<tr>
<td>CD</td>
<td>cardinal number</td>
<td>one, two, three</td>
<td>TO</td>
<td>&quot;to&quot;</td>
<td>to</td>
</tr>
<tr>
<td>DT</td>
<td>determiner</td>
<td>a, the</td>
<td>UH</td>
<td>interjection</td>
<td>ah, ops</td>
</tr>
<tr>
<td>EX</td>
<td>existential 'there'</td>
<td>there</td>
<td>VB</td>
<td>verb, base form</td>
<td>cat</td>
</tr>
<tr>
<td>FW</td>
<td>foreign word</td>
<td>mea culpa</td>
<td>VBD</td>
<td>verb, past tense</td>
<td>ate</td>
</tr>
<tr>
<td>IN</td>
<td>preposition/sub-conj</td>
<td>of, in, by</td>
<td>VBG</td>
<td>verb, gerund</td>
<td>eating</td>
</tr>
<tr>
<td>JJ</td>
<td>adjective</td>
<td>yellow</td>
<td>VBN</td>
<td>verb, past participle</td>
<td>eaten</td>
</tr>
<tr>
<td>JJR</td>
<td>adj. comparative</td>
<td>bigger</td>
<td>VBP</td>
<td>verb, non-3sg pres</td>
<td>eat</td>
</tr>
<tr>
<td>JJS</td>
<td>adj., superlative</td>
<td>wildest</td>
<td>VBZ</td>
<td>verb, 3sg pres</td>
<td>eats</td>
</tr>
<tr>
<td>LS</td>
<td>list item marker</td>
<td>1, 2, One</td>
<td>WDT</td>
<td>wh-determiner</td>
<td>which, that</td>
</tr>
<tr>
<td>MD</td>
<td>modal</td>
<td>can, should</td>
<td>WP</td>
<td>wh-pronoun</td>
<td>what, who</td>
</tr>
<tr>
<td>NN</td>
<td>noun, sing. or mass</td>
<td>Illiana</td>
<td>WPS</td>
<td>possessive wh-</td>
<td>whose</td>
</tr>
<tr>
<td>NNS</td>
<td>noun, plural</td>
<td>Illanias</td>
<td>WRB</td>
<td>wh-adverb</td>
<td>how, where</td>
</tr>
<tr>
<td>NNP</td>
<td>proper noun, singular</td>
<td>IBM</td>
<td>$</td>
<td>dollar sign</td>
<td>$</td>
</tr>
<tr>
<td>NNPS</td>
<td>proper noun, plural</td>
<td>Carolinas</td>
<td>#</td>
<td>pound sign</td>
<td>#</td>
</tr>
<tr>
<td>PDT</td>
<td>predeterminer</td>
<td>all, both</td>
<td>&quot;</td>
<td>left quote</td>
<td>‘ ‘ or ‘’</td>
</tr>
<tr>
<td>POS</td>
<td>possessive ending</td>
<td>‘s</td>
<td>&quot;</td>
<td>right quote</td>
<td>‘ ‘ or ‘’</td>
</tr>
<tr>
<td>PRP</td>
<td>personal pronoun</td>
<td>I, you, he</td>
<td>(</td>
<td>left parenthesis</td>
<td>[ { &lt;</td>
</tr>
<tr>
<td>PRPS</td>
<td>possessive pronoun</td>
<td>your, one’s</td>
<td>)</td>
<td>right parenthesis</td>
<td>] } &gt;</td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
<td>quickly, never</td>
<td>.</td>
<td>comma</td>
<td>:</td>
</tr>
<tr>
<td>RBX</td>
<td>adverb, comparative</td>
<td>faster</td>
<td>,</td>
<td>sentence-final punc</td>
<td>: ! ?</td>
</tr>
<tr>
<td>RBS</td>
<td>adverb, superlative</td>
<td>fastest</td>
<td>:</td>
<td>mid-sentence punc</td>
<td>: ; . -</td>
</tr>
</tbody>
</table>
Using the Penn Tagset

- The grand jury commented on a number of other topics.
Recall POS Tagging Difficulty

- Words often have more than one POS: back
  - The back door = JJ
  - On my back = NN
  - Win the voters back = RB
  - Promised to back the bill = VB

- The POS tagging problem is to determine the POS tag for a particular instance of a word.

These examples from Dekang Lin

How Hard is POS Tagging? Measuring Ambiguity

<table>
<thead>
<tr>
<th>Details</th>
<th>87-tag Original Brown</th>
<th>45-tag Treebank Brown</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Unambiguous (1 tag)</td>
<td>Ambiguous (2–7 tags)</td>
</tr>
<tr>
<td></td>
<td>44,019</td>
<td>5,490</td>
</tr>
<tr>
<td></td>
<td>38,857</td>
<td>8,844</td>
</tr>
<tr>
<td>2 tags</td>
<td>4,967</td>
<td>6,731</td>
</tr>
<tr>
<td>3 tags</td>
<td>411</td>
<td>1,621</td>
</tr>
<tr>
<td>4 tags</td>
<td>91</td>
<td>357</td>
</tr>
<tr>
<td>5 tags</td>
<td>17</td>
<td>90</td>
</tr>
<tr>
<td>6 tags</td>
<td>2 (well, beat)</td>
<td>32</td>
</tr>
<tr>
<td>7 tags</td>
<td>2 (still, down)</td>
<td>6 (well, set, round, open, fit, down)</td>
</tr>
<tr>
<td>8 tags</td>
<td></td>
<td>4 ('s, half, back, a)</td>
</tr>
<tr>
<td>9 tags</td>
<td></td>
<td>3 (that, more, in)</td>
</tr>
</tbody>
</table>
Tagging Whole Sentences with POS is Hard too

- Ambiguous POS contexts
  - E.g., Time flies like an arrow.
- Possible POS assignments
  - Time/[V,N] flies/[V,N] like/[V,Prep] an/Det arrow/N
  - Time/N flies/V like/Prep an/Det arrow/N
  - Time/V flies/N like/Prep an/Det arrow/N
  - Time/N flies/N like/V an/Det arrow/N
  - ..... 

How Do We Disambiguate POS?

- Many words have only one POS tag (e.g. is, Mary, smallest)
- Others have a single most likely tag (e.g. Dog is less used as a V)
- Tags also tend to co-occur regularly with other tags (e.g. Det, N)
- In addition to conditional probabilities of words P(w_i|w_{n-1}), we can look at POS likelihoods P(t_i|t_{n-1}) to disambiguate sentences and to assess sentence likelihoods
More and Better Features ➔ Feature-based tagger

- Can do surprisingly well just looking at a word by itself:
  - Word: the: the → DT
  - Lowercased word: Importantly: importantly → RB
  - Prefixes: unfathomable: un- → JJ
  - Suffixes: Importantly: -ly → RB
  - Capitalization: Meridian: CAP → NNP
  - Word shapes: 35-year: d-x → JJ

- Then build a maxent (or whatever) model to predict tag
  - Maxent P(t|w): 93.7% overall / 82.6% unknown

Overview: POS Tagging Accuracies

- Rough accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - Maxent P(t|w): 93.7% / 82.6%
  - Upper bound: ~98% (human)
How to improve supervised results?

- Build better features!
  
  PRP VBD IN RB IN PRP VBD .
  They left as soon as he arrived .

- We could fix this with a feature that looked at the next word
  JJ
  NNP NNS VBD VBN .
  Intrinsic flaws remained undetected .

- We could fix this by linking capitalized words to their lowercase versions

Tagging Without Sequence Information

<table>
<thead>
<tr>
<th>Model</th>
<th>Features</th>
<th>Token</th>
<th>Unknown</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>56,805</td>
<td>93.69%</td>
<td>82.61%</td>
<td>26.74%</td>
</tr>
<tr>
<td>3Words</td>
<td>239,767</td>
<td>96.57%</td>
<td>86.78%</td>
<td>48.27%</td>
</tr>
</tbody>
</table>

Using words only in a straight classifier works as well as a basic (HMM or discriminative) sequence model!!
For tagging, a change from generative to discriminative model does not by itself result in great improvement.

One profits from models for specifying dependence on overlapping features of the observation such as spelling, suffix analysis, etc.

Can allow integration of rich features of the observations, but can suffer strongly from assuming independence from following observations; this effect can be relieved by adding dependence on following words.

This additional power (of the MEMM, CRF, Perceptron models) has been shown to result in improvements in accuracy.

The higher accuracy of discriminative models comes at the price of much slower training.

---

**Rule-Based Tagging**

- Start with a dictionary
- Assign all possible tags to words from the dictionary
- Write rules by hand to selectively remove tags
- Leaving the correct tag for each word.
Start With a Dictionary

- she: PRP
- promised: VBN,VBD
- to TO
- back: VB, JJ, RB, NN
- the: DT
- bill: NN, VB
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
VBN JJ VB
PRP VBD TO VB DT NN
She promised to back the bill
POS Tagging as Sequence Classification

- We are given a sentence (an “observation” or “sequence of observations”)
  - *Secretariat is expected to race tomorrow*
- What is the best sequence of tags that corresponds to this sequence of observations?
- Probabilistic view:
  - Consider all possible sequences of tags
  - Out of this universe of sequences, choose the tag sequence which is most probable given the observation sequence of n words $w_1...w_n$.

Getting to HMMs (Hidden Markov Models)

- We want, out of all sequences of n tags $t_1...t_n$ the single tag sequence such that $P(t_1...t_n|w_1...w_n)$ is highest.

\[
\hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n|w_1^n)
\]

- Hat ^ means “our estimate of the best one”
- Argmax$_x f(x)$ means “the x such that f(x) is maximized”
Getting to HMMs

- This equation is guaranteed to give us the best tag sequence

\[ \hat{t}_1^n = \arg\max_{t_1^n} P(t_1^n | w_1^n) \]

- But how to make it operational? How to compute this value?

- Intuition of Bayesian classification:
  - Use Bayes rule to transform this equation into a set of other probabilities that are easier to compute

Using Bayes Rule

\[ P(x|y) = \frac{P(y|x)P(x)}{P(y)} \]

\[ \hat{t}_1^n = \arg\max_{t_1^n} \frac{P(w_1^n|t_1^n)P(t_1^n)}{P(w_1^n)} \]

\[ \hat{t}_1^n = \arg\max_{t_1^n} P(w_1^n|t_1^n)P(t_1^n) \]
Likelihood and Prior

\[ \hat{t}^n_1 = \arg \max_{t_1^n} \frac{P(w_1^n|t_1^n) \cdot P(t_1^n)}{P(t_1^n)} \]

\[ P(w_1^n|t_1^n) \approx \prod_{i=1}^{n} P(w_i|t_i) \]

\[ P(t_1^n) \approx \prod_{i=1}^{n} P(t_i|t_{i-1}) \]

\[ \hat{t}^n_i = \arg \max_{t_i^n} P(t_i^n|w_1^n) \approx \arg \max_{t_i^n} \prod_{i=1}^{n} P(w_i|t_i)P(t_i|t_{i-1}) \]

Two Kinds of Probabilities

- Tag transition probabilities \( p(t_i|t_{i-1}) \)
  - Determiners likely to precede adjs and nouns
    - That/DT flight/NN
    - The/DT yellow/JJ hat/NN
    - So we expect \( P(\text{NN}|\text{DT}) \) and \( P(\text{JJ}|\text{DT}) \) to be high
    - But \( P(\text{DT}|\text{JJ}) \) to be:
  - Compute \( P(\text{NN}|\text{DT}) \) by counting in a labeled corpus:
    \[ P(t_i|t_{i-1}) = \frac{C(t_{i-1}, t_i)}{C(t_{i-1})} \]
    \[ P(\text{NN}|\text{DT}) = \frac{C(\text{DT}, \text{NN})}{C(\text{DT})} = \frac{56,509}{116,454} = .49 \]
Two Kinds of Probabilities

- Word likelihood probabilities $p(w_i|t_i)$
  - VBZ (3sg Pres verb) likely to be “is”
  - Compute $P(is|VBZ)$ by counting in a labeled corpus:

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

$$P(is|VBZ) = \frac{C(VBZ, is)}{C(VBZ)} = \frac{10,073}{21,627} = .47$$

Example: The Verb “race”

- Secretariat/NNP is/VBZ expected/VBN to/TO race/VB tomorrow/NR
- People/NNS continue/VB to/TO inquire/VB the/DT reason/NN for/IN the/DT race/NN for/IN outer/JJ space/NN
- How do we pick the right tag?
Disambiguating “race”

Example

- $P(\text{NN}|\text{TO}) = .00047$
- $P(\text{VB}|\text{TO}) = .83$
- $P(\text{race}|\text{NN}) = .00057$
- $P(\text{race}|\text{VB}) = .00012$
- $P(\text{NR}|\text{VB}) = .0027$
- $P(\text{NR}|\text{NN}) = .0012$

- $P(\text{VB}|\text{TO})P(\text{NR}|\text{VB})P(\text{race}|\text{VB}) = .00000027$
- $P(\text{NN}|\text{TO})P(\text{NR}|\text{NN})P(\text{race}|\text{NN}) = .00000000032$

So we (correctly) choose the verb reading
Hidden Markov Models

- What we’ve described with these two kinds of probabilities is a Hidden Markov Model (HMM)

Definitions

- A weighted finite-state automaton adds probabilities to the arcs
  - The sum of the probabilities leaving any arc must sum to one
- A Markov chain is a special case of a WFSA in which the input sequence uniquely determines which states the automaton will go through
- Markov chains can’t represent inherently ambiguous problems
  - Useful for assigning probabilities to unambiguous sequences
Markov Chain for Weather

Weather continued
Markov Chain for Words

Markov Chain: “First-order observable Markov Model”

- A set of states
  - \( Q = q_1, q_2, ..., q_N \); the state at time \( t \) is \( q_t \)
- Transition probabilities:
  - A set of probabilities \( A = a_{01}a_{02}...a_{n1}...a_{nn} \).
  - Each \( a_{ij} \) represents the probability of transitioning from state \( i \) to state \( j \).
  - The set of these is the transition probability matrix \( A \)
- Current state only depends on previous state

\[
P(q_i \mid q_1...q_{i-1}) = P(q_i \mid q_{i-1})
\]
Markov Chain for Weather

- What is the probability of 4 consecutive rainy days?
- Sequence is rainy-rainy-rainy-rainy
- I.e., state sequence is 3-3-3-3
- \( P(3,3,3,3) = \)

\[
\pi_3 a_{33} a_{33} a_{33} = 0.2 \times (0.6)^3 = 0.0432
\]
**HMM for Ice Cream**

- You are a climatologist in the year 2799
- Studying global warming
- You can’t find any records of the weather in Pittsburgh for summer of 2016
- But you find a diary
- Which lists how many ice-creams someone ate every date that summer
- Our job: figure out how hot it was

**Hidden Markov Model**

- For Markov chains, the output symbols are the same as the states.
  - See **hot** weather: we’re in state **hot**
- But in part-of-speech tagging (and other things)
  - The output symbols are **words**
  - But the hidden states are **part-of-speech tags**
- So we need an extension!
- A Hidden Markov Model is an extension of a Markov chain in which the input symbols are not the same as the states.
- This means we don’t know which state we are in.
Hidden Markov Models

- States $Q = q_1, q_2... q_N$;
- Observations $O= o_1, o_2... o_N$;
  - Each observation is a symbol from a vocabulary $V = \{v_1, v_2... v_V\}$
- Transition probabilities
  - Transition probability matrix $A = \{a_{ij}\}$
    $$a_{ij} = P(q_t = j \mid q_{t-1} = i) \quad 1 \leq i, j \leq N$$
- Observation likelihoods
  - Output probability matrix $B=\{b_i(k)\}$
    $$b_i(k) = P(X_t = o_k \mid q_t = i)$$
    $$\pi_i = P(q_1 = i) \quad 1 \leq i \leq N$$
- Special initial probability vector $\pi$

Task

- Given
  - Ice Cream Observation Sequence:
    1,2,3,2,2,2,3...
- Produce:
Weather/Ice Cream HMM

- Hidden States: \{Hot, Cold\}
- Transition probabilities (A Matrix) between H and C
- Observations: \{1, 2, 3\} # of ice creams eaten per day
Back to POS Tagging: Transition Probabilities

Observation Likelihoods
What can HMMs Do?

- **Likelihood**: Given an HMM $\lambda$ and an observation sequence $O$, determine the likelihood $P(O, \lambda)$: *language modeling*

- **Decoding**: Given an observation sequence $O$ and an HMM $\lambda$, discover the best hidden state sequence $Q$: Given seq of ice creams, what was the most likely weather on those days? *(tagging)*

- **Learning**: Given an observation sequence $O$ and the set of states in the HMM, learn the HMM *parameters*

Decoding

- Ok, now we have a complete model that can give us what we need. Recall that we need to get

$$\hat{t}_1^m = \arg\max_{t_1^n} P(t_1^n | w_1^n)$$

- We could just enumerate all paths given the input and use the model to assign probabilities to each.
  - Not a good idea.
  - In practice: Viterbi Algorithm (dynamic programming)
Viterbi Algorithm

- Intuition: since state transition out of a state only depend on the current state (and not previous states), we can record for each state the optimal path
- We record
  - Cheapest cost to state at step
  - Backtrace for that state to best predecessor

Viterbi Summary

- Create an array
  - With columns corresponding to inputs
  - Rows corresponding to possible states
- Sweep through the array in one pass filling the columns left to right using our transition probs and observations probs
- Dynamic programming key is that we need only store the MAX prob path to each cell (not all paths).
Viterbi Example

Another Viterbi Example

- Analyzing “Fish sleep”
  - Done in class
Evaluation

- So once you have your POS tagger running how do you evaluate it?
  - Overall error rate with respect to a gold-standard test set.
  - Error rates on particular tags
  - Error rates on particular words
  - Tag confusions...
- Need a baseline – just the most frequent tag is 90% accurate!

Error Analysis

- Look at a confusion matrix

<table>
<thead>
<tr>
<th></th>
<th>IN</th>
<th>JJ</th>
<th>NN</th>
<th>NNP</th>
<th>RB</th>
<th>VBD</th>
<th>VBN</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN</td>
<td></td>
<td>.2</td>
<td></td>
<td></td>
<td>.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JJ</td>
<td>.2</td>
<td></td>
<td>3.3</td>
<td>2.1</td>
<td>1.7</td>
<td>.2</td>
<td>2.7</td>
</tr>
<tr>
<td>NN</td>
<td>8.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>.2</td>
</tr>
<tr>
<td>NNP</td>
<td>.2</td>
<td>3.3</td>
<td>4.1</td>
<td></td>
<td>.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VBD</td>
<td>.3</td>
<td>.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4.4</td>
</tr>
<tr>
<td>VBN</td>
<td>2.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- See what errors are causing problems
  - 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).
- Important: 100% is impossible even for human annotators.

More Complex Issues

- Tag indeterminacy: when ‘truth’ isn’t clear
  Caribbean cooking, child seat
- Tagging multipart words
  wouldn’t --> would/MD n’t/RB
- How to handle unknown words
  - Assume all tags equally likely
  - Assume same tag distribution as all other singletons in corpus
  - Use morphology, word length,....
Other Tagging Tasks

- Noun Phrase (NP) Chunking
  - [the student] said [the exam] is hard
- Three tabs
  - B = beginning of NP
  - I = continuing in NP
  - O = other word
- Tagging result
  - The/B student/I said/O the/B exam/I is/0 hard/0

Summary

- Parts of speech
- Tagsets
- Part of speech tagging
- Rule-Based, HMM Tagging
- Optional: Hybrid (2nd edition)
- Optional: MEMM (3rd edition)
Transformation-Based (Brill) Tagging (Optional)

- Combines Rule-based and Stochastic Tagging
  - Like rule-based because rules are used to specify tags in a certain environment
  - Like stochastic approach because we use a tagged corpus to find the best performing rules
    - Rules are learned from data
- Input:
  - Tagged corpus
  - Dictionary (with most frequent tags)

Transformation-Based Tagging

- Basic Idea: Strip tags from tagged corpus and try to learn them by rule application
  - For untagged, first initialize with most probable tag for each word
  - Change tags according to best rewrite rule, e.g. "if word-1 is a determiner and word is a verb then change the tag to noun"
  - Compare to gold standard
  - Iterate
- Rules created via rule templates, e.g. of the form if word-1 is an X and word is a Y then change the tag to Z"
  - Find rule that applies correctly to most tags and apply
  - Iterate on newly tagged corpus until threshold reached
  - Return ordered set of rules
- NB: Rules may make errors that are corrected by later rules
Templates for TBL

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.

<table>
<thead>
<tr>
<th>#</th>
<th>Change tags From</th>
<th>To</th>
<th>Condition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NN</td>
<td>VB</td>
<td>Previous tag is TO</td>
<td>is/VBZ expected/VBN to/TO race/NN tomorrow/NN</td>
</tr>
<tr>
<td>2</td>
<td>VBP</td>
<td>VB</td>
<td>One of the previous 3 tags is MD</td>
<td>might/MD vanish/VBP to/TO VBN</td>
</tr>
<tr>
<td>3</td>
<td>NN</td>
<td>VB</td>
<td>One of the previous 2 tags is MD</td>
<td>might/MD not really/NN to/TO VB</td>
</tr>
<tr>
<td>4</td>
<td>VB</td>
<td>NN</td>
<td>One of the previous 2 tags is DT</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>VBD</td>
<td>VBN</td>
<td>One of the previous 3 tags is VBZ</td>
<td></td>
</tr>
</tbody>
</table>

Sample TBL Rule Application

- Labels every word with its most-likely tag
  - E.g. *race* occurrences in the Brown corpus:
    - $P(\text{NN/race}) = .98$
    - $P(\text{VB/race}) = .02$
    - *is/VBZ expected/VBN to/TO race/NN tomorrow/NN*
  - Then TBL applies the following rule
    - “Change NN to VB when previous tag is TO”
    - *is/VBZ expected/VBN to/TO race/NN tomorrow/NN*
    - becomes
    - ...

1/26/2017 87

1/26/2017 88
Sample TBL Rule Application

- Labels every word with its most-likely tag
  - E.g. `race` occurrences in the Brown corpus:
    - $P(\text{NN}/\text{race}) = .98$
    - $P(\text{VB}/\text{race}) = .02$
    - is/\text{VBZ} expected/\text{VBN} to/\text{TO} race/\text{NN} tomorrow/\text{NN}$

- Then TBL applies the following rule
  - “Change NN to VB when previous tag is TO”
    - ... is/\text{VBZ} expected/\text{VBN} to/\text{TO} race/\text{NN} tomorrow/\text{NN}$
    - becomes
      - ... is/\text{VBZ} expected/\text{VBN} to/\text{TO} race/\text{VB} tomorrow/\text{NN}$

TBL Tagging Algorithm

- Step 1: Label every word with most likely tag (from dictionary)
- Step 2: Check every possible transformation & select one which most improves tag accuracy (cf Gold)
- Step 3: Re-tag corpus applying this rule, and add rule to end of rule set
- Repeat 2-3 until some stopping criterion is reached, e.g., X% correct with respect to training corpus
- RESULT: Ordered set of transformation rules to use on new data tagged only with most likely POS tags
TBL Issues

- Problem: Could keep applying (new) transformations ad infinitum
- Problem: Rules are learned in ordered sequence
- Problem: Rules may interact
- But: Rules are compact and can be inspected by humans