CSP 517
Natural Language Processing
Winter 2015

Parts of Speech

Yejin Choi

[Slides adapted from Dan Klein, Luke Zettlemoyer]
Overview

- POS Tagging
- Feature Rich Techniques
  - Maximum Entropy Markov Models (MEMMs)
  - Structured Perceptron
  - Conditional Random Fields (CRFs)
One basic kind of linguistic structure: syntactic word classes

<table>
<thead>
<tr>
<th>Open class (lexical) words</th>
<th>Closed class (functional)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Nouns</strong></td>
<td><strong>Determiners</strong></td>
</tr>
<tr>
<td>Proper</td>
<td>the some</td>
</tr>
<tr>
<td>IBM</td>
<td></td>
</tr>
<tr>
<td>Italy</td>
<td></td>
</tr>
<tr>
<td>Common</td>
<td><strong>Conjunctions</strong></td>
</tr>
<tr>
<td>cat / cats</td>
<td>and or</td>
</tr>
<tr>
<td>snow</td>
<td></td>
</tr>
<tr>
<td><strong>Verbs</strong></td>
<td><strong>Modals</strong></td>
</tr>
<tr>
<td>Main</td>
<td>can</td>
</tr>
<tr>
<td>see</td>
<td>had</td>
</tr>
<tr>
<td>registered</td>
<td></td>
</tr>
<tr>
<td><strong>Adjectives</strong></td>
<td></td>
</tr>
<tr>
<td>yellow</td>
<td></td>
</tr>
<tr>
<td><strong>Adverbs</strong></td>
<td></td>
</tr>
<tr>
<td>slowly</td>
<td></td>
</tr>
<tr>
<td><strong>Numbers</strong></td>
<td></td>
</tr>
<tr>
<td>122,312</td>
<td></td>
</tr>
<tr>
<td>one</td>
<td></td>
</tr>
<tr>
<td>... more</td>
<td></td>
</tr>
<tr>
<td><strong>Prepositions</strong></td>
<td></td>
</tr>
<tr>
<td>to with</td>
<td></td>
</tr>
<tr>
<td><strong>Particles</strong></td>
<td></td>
</tr>
<tr>
<td>off up</td>
<td></td>
</tr>
<tr>
<td>... more</td>
<td></td>
</tr>
</tbody>
</table>
Penn Treebank POS: 36 possible tags, 34 pages of tagging guidelines.

<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
</table>
| CC  | conjunction, coordinating          | and both but either or  
| CD  | numeral, cardinal                  | mid-1890 nine-thirty 0.5 one  
| DT  | determiner                         | a all an every no that the there  
| EX  | existential there                   | gemeinschaft hund ich jeux  
| FW  | foreign word                       | among whether out on by if  
| IN  | preposition or conjunction, subordinating | third ill-mannered regrettable  
| JJ  | adjective or numeral, ordinal      | braver cheaper taller  
| JJR | adjective, comparative             | bravest cheapest tallest  
| JJS | adjective, superlative             | can may might will would  
| MD  | modal auxiliary                    | cabbage thermostat investment subhumanity  
| NN  | noun, common, singular or mass      | Motown Cougar Yvette Liverpool  
| NNP | noun, proper, singular              | Americans Materials States  
| NNPS| noun, proper, plural               | undergraduates bric-a-brac averages  
| NNS | noun, common, plural               | 's  
| POS | genitive marker                    | hers himself it we them  
| PRP | pronoun, personal                  | her his mine my our ours their thy your  
| PRP$| pronoun, possessive                | occasionally maddeningly adventurously  
| RB  | adverb                             | further gloomier heavier less-perfectly  
| RBR | adverb, comparative                | best biggest nearest worst  
| RBS | adverb, superlative                | aboard away back by on open through  

"to" as preposition or infinitive
<table>
<thead>
<tr>
<th>Tag</th>
<th>Description</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>PRP</td>
<td>pronoun, personal</td>
<td>hers, himself, it, we, them</td>
</tr>
<tr>
<td>PRP$</td>
<td>pronoun, possessive</td>
<td>her, his, mine, my, our, ours</td>
</tr>
<tr>
<td>RB</td>
<td>adverb</td>
<td>occasionally, maddeningly, adventurously</td>
</tr>
<tr>
<td>RBR</td>
<td>adverb, comparative</td>
<td>further, gloomier, heavier</td>
</tr>
<tr>
<td>RBS</td>
<td>adverb, superlative</td>
<td>best, biggest, nearest, worst</td>
</tr>
<tr>
<td>RP</td>
<td>particle</td>
<td>aboard, away, back, by, on, open through</td>
</tr>
<tr>
<td>TO</td>
<td>&quot;to&quot; as preposition or infinitive marker</td>
<td>to</td>
</tr>
<tr>
<td>UH</td>
<td>interjection</td>
<td>huh, howdy, uh, whammo, shucks, heck</td>
</tr>
<tr>
<td>VB</td>
<td>verb, base form</td>
<td>ask, bring, fire, see, take</td>
</tr>
<tr>
<td>VBD</td>
<td>verb, past tense</td>
<td>pleaded, swiped, registered, saw</td>
</tr>
<tr>
<td>VBG</td>
<td>verb, present participle or gerund</td>
<td>stirring, focusing, approaching, erasing</td>
</tr>
<tr>
<td>VBN</td>
<td>verb, past participle</td>
<td>dilapidated, imitated, reunified, unsettled</td>
</tr>
<tr>
<td>VBP</td>
<td>verb, present tense, not 3rd person</td>
<td>twist, appear, comprise, mold, postpone</td>
</tr>
<tr>
<td>VBZ</td>
<td>verb, present tense, 3rd person</td>
<td>bases, reconstructs, marks, uses</td>
</tr>
<tr>
<td>WDT</td>
<td>WH-determiner</td>
<td>that, what, whatever, which, whichever</td>
</tr>
<tr>
<td>WP</td>
<td>WH-pronoun</td>
<td>that, what, whatever, which, who, whom</td>
</tr>
<tr>
<td>WP$</td>
<td>WH-pronoun, possessive</td>
<td>whose, however, whenever, where, why</td>
</tr>
<tr>
<td>WRB</td>
<td>Wh-adverb</td>
<td></td>
</tr>
</tbody>
</table>

Part-of-Speech Ambiguity

- Words can have multiple parts of speech

Fed raises interest rates 0.5 percent

Mrs./NNP Shaefer/NNP never/RB got/VBD around/RP to/TO joining/VBG
All/DTD we/PRP gotta/VBN do/VB is/VBZ go/VB around/IN the/DTD corner/NN
Chateau/NNP Petrus/NNP costs/VBZ around/RB 250/CD

- Two basic sources of constraint:
  - Grammatical environment
  - Identity of the current word

- Many more possible features:
  - Suffixes, capitalization, name databases (gazetteers), etc…
Why POS Tagging?

- **Useful in and of itself (more than you’d think)**
  - Text-to-speech: record, lead
  - Lemmatization: saw\[v\] → see, saw\[n\] → saw
  - Quick-and-dirty NP-chunk detection: grep \{JJ | NN\}* \{NN | NNS\}

- **Useful as a pre-processing step for parsing**
  - Less tag ambiguity means fewer parses
  - However, some tag choices are better decided by parsers

The average of interbank **offered** rates plummeted …

```
  IN
DT  NNP  NN  VBD  VBN  RP  NN  NNS
The Georgia branch had taken on loan commitments …
```

```
  VDN
DT  NN  IN  NN  VBD  NNS  VBD
The average of interbank **offered** rates plummeted …
```
Baselines and Upper Bounds

- Choose the most common tag
  - 90.3% with a bad unknown word model
  - 93.7% with a good one

- Noise in the data
  - Many errors in the training and test corpora
  - Probably about 2% guaranteed error from noise (on this data)
Ambiguity in POS Tagging

- **Particle (RP) vs. preposition (IN)**
  - He talked *over* the deal.
  - He talked *over* the telephone.

- **past tense (VBD) vs. past participle (VBN)**
  - The horse *walked* past the barn.
  - The horse *walked* past the barn fell.

- **noun vs. adjective?**
  - The *executive* decision.

- **noun vs. present participle**
  - *Fishing* can be fun
Ambiguity in POS Tagging

- “Like” can be a verb or a preposition
  - I like/VBP candy.
  - Time flies like/IN an arrow.

- “Around” can be a preposition, particle, or adverb
  - I bought it at the shop around/IN the corner.
  - I never got around/RP to getting a car.
  - A new Prius costs around/RB $25K.
Overview: Accuracies

- **Roadmap of (known / unknown) accuracies:**
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%

- **TnT (Brants, 2000):**
  - A carefully smoothed trigram tagger
  - Suffix trees for emissions
  - 96.7% on WSJ text (SOA is ~97.5%)

- Upper bound: ~98%

Most errors on unknown words
Common Errors

- Common errors [from Toutanova & Manning 00]

<table>
<thead>
<tr>
<th>JJ</th>
<th>NN</th>
<th>NNP</th>
<th>NNPS</th>
<th>RB</th>
<th>RP</th>
<th>IN</th>
<th>VB</th>
<th>VBD</th>
<th>VBN</th>
<th>VBP</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>JJ</td>
<td>0</td>
<td>177</td>
<td>56</td>
<td>0</td>
<td>61</td>
<td>2</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>108</td>
<td>0</td>
</tr>
<tr>
<td>NN</td>
<td>244</td>
<td>0</td>
<td>103</td>
<td>0</td>
<td>12</td>
<td>1</td>
<td>1</td>
<td>29</td>
<td>5</td>
<td>6</td>
<td>19</td>
</tr>
<tr>
<td>NNP</td>
<td>107</td>
<td>106</td>
<td>0</td>
<td>132</td>
<td>5</td>
<td>0</td>
<td>7</td>
<td>5</td>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>NNPS</td>
<td>1</td>
<td>0</td>
<td>110</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>RB</td>
<td>72</td>
<td>21</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>16</td>
<td>138</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>RP</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>39</td>
<td>0</td>
<td>65</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>IN</td>
<td>11</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>169</td>
<td>103</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>VB</td>
<td>17</td>
<td>64</td>
<td>9</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>7</td>
<td>85</td>
</tr>
<tr>
<td>VBD</td>
<td>10</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>143</td>
<td>2</td>
</tr>
<tr>
<td>VBN</td>
<td>101</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>108</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>VBP</td>
<td>5</td>
<td>34</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>49</td>
<td>6</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Total</td>
<td>626</td>
<td>536</td>
<td>348</td>
<td>144</td>
<td>317</td>
<td>122</td>
<td>279</td>
<td>102</td>
<td>140</td>
<td>269</td>
<td>108</td>
</tr>
</tbody>
</table>

NN/JJ | NN | VBD RP/IN DT NN | RB | VBD/VBN | NNS

official knowledge | made up the story | recently sold shares
What about better features?

- Choose the most common tag
  - 90.3% with a bad unknown word model
  - 93.7% with a good one

- What about looking at a word and its environment, but no sequence information?
  - Add in previous / next word: the __
  - Previous / next word shapes: X __ X
  - Occurrence pattern features: [X: x X occurs]
  - Crude entity detection: __ ...... (Inc.|Co.)
  - Phrasal verb in sentence?: put _______
  - Conjunctions of these things

- Uses lots of features: > 200K
Overview: Accuracies

- Roadmap of (known / unknown) accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - TnT (HMM++): 96.2% / 86.0%
  - Maxent $P(s_i|x)$: 96.8% / 86.8%

- Q: What does this say about sequence models?
- Q: How do we add more features to our sequence models?

- Upper bound: ~98%
MEMM Taggers

- **One step up:** also condition on previous tags

\[
p(s_1 \ldots s_m | x_1 \ldots x_m) = \prod_{i=1}^{m} p(s_i | s_1 \ldots s_{i-1}, x_1 \ldots x_m)
\]

\[
= \prod_{i=1}^{m} p(s_i | s_{i-1}, x_1 \ldots x_m)
\]

- Train up \(p(s_i | s_{i-1}, x_1 \ldots x_m)\) as a discrete log-linear (maxent) model, then use to score sequences

\[
p(s_i | s_{i-1}, x_1 \ldots x_m) = \frac{\exp \left( w \cdot \phi(x_1 \ldots x_m, i, s_{i-1}, s_i) \right)}{\sum_{s'} \exp \left( w \cdot \phi(x_1 \ldots x_m, i, s_{i-1}, s') \right)}
\]

- This is referred to as an MEMM tagger [Ratnaparkhi 96]
- Beam search effective! (Why?)
- What’s the advantage of beam size 1?
The HMM State Lattice / Trellis (repeat slide)

\[
\begin{array}{cccccccc}
& ^ & ^ & ^ & ^ & ^ & ^ & ^ \\
\hat{N} & N & N & N & N & N & N & N \\
V & V & V & V & V & V & V & V \\
D & D & D & D & D & D & D & D \\
$ & $ & $ & $ & $ & $ & $ & $ \\
\text{START} & \text{Fed} & \text{raises} & \text{interest} & \text{rates} & \text{STOP} \\
\end{array}
\]
The MEMM State Lattice / Trellis

x = START       Fed           raises       interest         rates         STOP
Decoding

- **Decoding maxent taggers:**
  - Just like decoding HMMs
  - Viterbi, beam search, posterior decoding

- **Viterbi algorithm (HMMs):**
  - Define $\pi(i, s_i)$ to be the max score of a sequence of length $i$ ending in tag $s_i$
  $$
  \pi(i, s_i) = \max_{s_{i-1}} e(x_i | s_i)q(s_i | s_{i-1})\pi(i - 1, s_{i-1})
  $$

- **Viterbi algorithm (Maxent):**
  - Can use same algorithm for MEMMs, just need to redefine $\pi(i, s_i)$!
  $$
  \pi(i, s_i) = \max_{s_{i-1}} p(s_i | s_{i-1}, x_1 \ldots x_m)\pi(i - 1, s_{i-1})
  $$
Overview: Accuracies

- Roadmap of (known / unknown) accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - TnT (HMM++): 96.2% / 86.0%
  - Maxent $P(s_i|x)$: 96.8% / 86.8%
  - MEMM tagger: 96.9% / 86.9%

- Upper bound: ~98%
Global Discriminative Taggers

- Newer, higher-powered discriminative sequence models
  - CRFs (also perceptrons, M3Ns)
  - Do not decompose training into independent local regions
  - Can be deathly slow to train – require repeated inference on training set

- Differences can vary in importance, depending on task
- However: one issue worth knowing about in local models
  - “Label bias” and other explaining away effects
  - MEMM taggers’ local scores can be near one without having both good “transitions” and “emissions”
  - This means that often evidence doesn’t flow properly
  - Why isn’t this a big deal for POS tagging?
  - Also: in decoding, condition on predicted, not gold, histories
Linear Models: Perceptron

- The perceptron algorithm
  - Iteratively processes the training set, reacting to training errors
  - Can be thought of as trying to drive down training error

- The (online) perceptron algorithm:
  - Start with zero weights
  - Visit training instances \((x_i, y_i)\) one by one
    - Make a prediction
      \[ y^* = \arg \max_y w \cdot \phi(x_i, y) \]
    - If correct \((y^* = y_i)\): no change, goto next example!
    - If wrong: adjust weights
      \[ w = w + \phi(x_i, y_i) - \phi(x_i, y^*) \]

Challenge: How to compute \(\arg \max\) efficiently?
Decoding

- **Linear Perceptron**
  
  \[ s^* = \arg \max_s w \cdot \Phi(x, s) \cdot \theta \]

  - Features must be local, for \( x=x_1\ldots x_m \), and \( s=s_1\ldots s_m \)

  \[ \Phi(x, s) = \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_j) \]
The MEMM State Lattice / Trellis (repeat)

x = START       Fed           raises       interest         rates         STOP
The Perceptron State Lattice / Trellis

\[ x = \text{START} \quad \text{Fed} \quad \text{raises} \quad \text{interest} \quad \text{rates} \quad \text{STOP} \]
Decoding

- **Linear Perceptron**
  \[
  s^* = \arg \max_s w \cdot \Phi(x, s) \cdot \theta
  \]
  - Features must be local, for \( x = x_1 \ldots x_m \), and \( s = s_1 \ldots s_m \)
  \[
  \Phi(x, s) = \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_j)
  \]
  - Define \( \pi(i, s_i) \) to be the max score of a sequence of length \( i \) ending in tag \( s_i \)
  \[
  \pi(i, s_i) = \max_{s_{i-1}} w \cdot \phi(x, i, s_{i-1}, s_i) + \pi(i - 1, s_{i-1})
  \]

- **Viterbi algorithm (HMMs):**
  \[
  \pi(i, s_i) = \max_{s_{i-1}} e(x_i | s_i) q(s_i | s_{i-1}) \pi(i - 1, s_{i-1})
  \]

- **Viterbi algorithm (Maxent):**
  \[
  \pi(i, s_i) = \max_{s_{i-1}} p(s_i | s_{i-1}, x_1 \ldots x_m) \pi(i - 1, s_{i-1})
  \]
Overview: Accuracies

- Roadmap of (known / unknown) accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - TnT (HMM++): 96.2% / 86.0%
  - Maxent P(s_i|x): 96.8% / 86.8%
  - MEMM tagger: 96.9% / 86.9%
  - Perceptron: 96.7% / ??
  - Upper bound: ~98%
Conditional Random Fields (CRFs) [Lafferty, McCallum, Pereira 01]

- Maximum entropy (logistic regression)

\[
p(y|x; w) = \frac{\exp \left( w \cdot \phi(x, y) \right)}{\sum_{y'} \exp \left( w \cdot \phi(x, y') \right)}
\]

- Learning: maximize the (log) conditional likelihood of training data \( \{(x_i, y_i)\}_{i=1}^{n} \)

\[
\frac{\partial}{\partial w_j} L(w) = \sum_{i=1}^{n} \left( \phi_j(x_i, y_i) - \sum_{y} p(y|x_i; w) \phi_j(x_i, y) \right) - \lambda w_j
\]

- Computational Challenges?
  - Most likely tag sequence, normalization constant, gradient
Decoding

- **CRFs**
  - Features must be local, for $x=x_1 \ldots x_m$, and $s=s_1 \ldots s_m$

$$p(s|x; w) = \frac{\exp (w \cdot \Phi(x, s))}{\sum_{s'} \exp (w \cdot \Phi(x, s'))}$$

$$\Phi(x, s) = \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_j)$$

$$\arg \max_s \frac{\exp (w \cdot \Phi(x, s))}{\sum_{s'} \exp (w \cdot \Phi(x, s'))} = \arg \max_s \exp (w \cdot \Phi(x, s))$$

$$= \arg \max_s w \cdot \Phi(x, s)$$

- **Same as Linear Perceptron!!!**

$$\pi(i, s_i) = \max_{s_{i-1}} \phi(x, i, s_{i-1}, s_i) + \pi(i - 1, s_{i-1})$$
CRFs: Computing Normalization*

\[ p(s|x; w) = \frac{\exp (w \cdot \Phi(x, s))}{\sum_{s'} \exp (w \cdot \Phi(x, s'))} \]

\[ \Phi(x, s) = \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_j) \]

\[ \sum_{s'} \exp (w \cdot \Phi(x, s')) = \sum_{s'} \exp \left( \sum_{j} w \cdot \phi(x, j, s_{j-1}, s_j) \right) \]

\[ = \sum_{s'} \prod_{j} \exp (w \cdot \phi(x, j, s_{j-1}, s_j)) \]

Define \( \text{norm}(i,s_i) \) to sum of scores for sequences ending in position \( i \)

\[ \text{norm}(i, y_i) = \sum_{s_{i-1}} \exp (w \cdot \phi(x, i, s_{i-1}, s_i)) \text{norm}(i - 1, s_{i-1}) \]

- **Forward Algorithm! Remember HMM case:**

\[ \alpha(i, y_i) = \sum_{y_{i-1}} e(x_i|y_i)q(y_i|y_{i-1})\alpha(i - 1, y_{i-1}) \]

- Could also use backward?
CRFs: Computing Gradient*  

\[
p(s|x; w) = \frac{\exp(w \cdot \Phi(x, s))}{\sum_{s'} \exp(w \cdot \Phi(x, s'))}
\]

\[
\Phi(x, s) = \sum_{j=1}^{m} \phi(x, j, s_{j-1}, s_j)
\]

\[
\frac{\partial}{\partial w_j} L(w) = \sum_{i=1}^{n} \left( \Phi_j(x_i, s_i) - \sum_{s} p(s|x_i; w) \Phi_j(x_i, s) \right) - \lambda w_j
\]

\[
\sum_{s} p(s|x_i; w) \Phi_j(x_i, s) = \sum_{s} p(s|x_i; w) \sum_{j=1}^{m} \phi_k(x_i, j, s_{j-1}, s_j)
\]

\[
= \sum_{j=1}^{m} \sum_{a,b} \sum_{s:s_{j-1}=a,s_b=b} p(s|x_i; w) \phi_k(x_i, j, s_{j-1}, s_j)
\]

- Need forward and backward messages  
  See notes for full details!
Overview: Accuracies

Roadmap of (known / unknown) accuracies:

- Most freq tag: ~90% / ~50%
- Trigram HMM: ~95% / ~55%
- TnT (HMM++): 96.2% / 86.0%
- Maxent P(s_i|x): 96.8% / 86.8%
- MEMM tagger: 96.9% / 86.9%
- Perceptron: 96.7% / ??
- CRF (untuned): 95.7% / 76.2%

Upper bound: ~98%
Cyclic Network

- Train two MEMMs, multiple together to score
- And be very careful
  - Tune regularization
  - Try lots of different features
  - See paper for full details

- [Toutanova et al 03]

(a) Left-to-Right CMM

(b) Right-to-Left CMM

(c) Bidirectional Dependency Network
Overview: Accuracies

- Roadmap of (known / unknown) accuracies:
  - Most freq tag: ~90% / ~50%
  - Trigram HMM: ~95% / ~55%
  - TnT (HMM++): 96.2% / 86.0%
  - Maxent P(s_i|x): 96.8% / 86.8%
  - MEMM tagger: 96.9% / 86.9%
  - Perceptron: 96.7% / ??
  - CRF (untuned): 95.7% / 76.2%
  - Cyclic tagger: 97.2% / 89.0%
  - Upper bound: ~98%
Domain Effects

- Accuracies degrade outside of domain
  - Up to triple error rate
  - Usually make the most errors on the things you care about in the domain (e.g. protein names)

- Open questions
  - How to effectively exploit unlabeled data from a new domain (what could we gain?)
  - How to best incorporate domain lexica in a principled way (e.g. UMLS specialist lexicon, ontologies)