CS388: Natural Language Processing

Lecture 15: HMMs, POS

Greg Durrett

The University of Texas at Austin
Administrivia

- Project 3 due Thursday
- No class Thursday, Greg’s office hours canceled Thursday (on for Wednesday)
Where are we in the course?

› So far, we have studied Transformer language models, which form representations of text *latently* via embeddings (using self attention + feedforward layers)

› So far, we have studied Transformer language models, which form representations of text *latently* via embeddings (using self attention + feedforward layers)
Language is Ambiguous!

From the start of the semester: “Teacher Strikes Idle Kids”

You
What would be some possible consequences of the events in an article with the following headline: "Teacher Strikes Idle Kids"?

ChatGPT
An article with the headline "Teacher Strikes Idle Kids" suggests a situation where teacher strikes have led to school closures, leaving students without formal education and supervision. Such events can have multiple consequences, affecting various stakeholders. Here are some possible consequences:

1. **Educational Setbacks**: Students may face delays in learning, potentially impacting their academic progress. Extended strikes can lead to gaps in knowledge, difficulties in catching up, and disruptions in the academic calendar.

2. **Social and Emotional Impact**: School provides a structured environment for...
Where are we in the course?

- So far, we have studied Transformer language models, which form representations of text *latently* via embeddings (using self attention + feedforward layers)

- Many discrete decisions about the structure of language are made in Transformer models

- How can we have a symbolic system (+ models) for representing what those decisions look like?
Where are we in the course?

- Next three lectures: structured prediction. Produce representations of language as sequences and trees.

- Language has hierarchical structure:

  - Understanding syntax fundamentally requires trees — the sentences have the same shallow analysis. But the first step we’ll take towards understanding this is understanding parts of speech.
This Lecture

- Part-of-speech tagging
- Hidden Markov Models, parameter estimation
- Viterbi algorithm
- POS taggers
- NER, CRFs, state-of-the-art in sequence modeling
POS Tagging
POS Tagging

Open class (lexical) words

Nouns
- Proper
  - IBM
  - Italy
- Common
  - cat / cats
  - snow

Verbs
- Main
  - see
  - registered
- Auxiliary
  - can
  - had

Adjectives
- yellow

Adverbs
- slowly

Numbers
- 122,312
- one

Prepositions
- to with

Particles
- off up

Closed class (functional)

Determiners
- the
- some

Conjunctions
- and
- or

Pronouns
- he
- its

... more

... more

Slide credit: Dan Klein
Fed raises interest rates 0.5 percent

Fed raises interest rates 0.5 percent

I hereby increase interest rates 0.5%

I’m 0.5% interested in the Fed’s raises!

‣ Other paths are also plausible but even more semantically weird...

‣ What governs the correct choice? Word + context
  ▸ Word identity: most words have <=2 tags, many have one (percent, the)
  ▸ Context: nouns start sentences, nouns follow verbs, etc.
Hidden Markov Models
Hidden Markov Models

- Input $x = (x_1, \ldots, x_n)$  
  Output $y = (y_1, \ldots, y_n)$

- Model the sequence of tags $y$ over words $x$ as a Markov process

- Markov property: future is conditionally independent of the past given the present

$$P(y_3|y_1, y_2) = P(y_3|y_2)$$

- If $y$ are tags, this roughly corresponds to assuming that the next tag only depends on the current tag, not anything before
Hidden Markov Models

- Input $x = (x_1, ..., x_n)$  
- Output $y = (y_1, ..., y_n)$ 
- $y \in T =$ set of possible tags (including STOP); 
- $x \in V =$ vocab of words

$P(y, x) = P(y_1) \prod_{i=2}^{n} P(y_i | y_{i-1}) \prod_{i=1}^{n} P(x_i | y_i)$

- Observation ($x$) depends only on current state ($y$)
HMMs: Parameters

- Input $x = (x_1, \ldots, x_n)$  Output $y = (y_1, \ldots, y_n)$

$$P(y, x) = P(y_1) \prod_{i=2}^{n} P(y_i|y_{i-1}) \prod_{i=1}^{n} P(x_i|y_i)$$

- Initial distribution: $|T| \times 1$ vector (distribution over initial states)
- Emission distribution: $|T| \times |V|$ matrix (distribution over words per tag)
- Transition distribution: $|T| \times |T|$ matrix (distribution over next tags per tag)
HMMs Example

Tags = \{N, V, STOP\}

Vocabulary = \{they, can, fish\}

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>STOP</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>V</th>
<th>STOP</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>1/5</td>
<td>3/5</td>
<td>1/5</td>
</tr>
<tr>
<td>V</td>
<td>1/5</td>
<td>1/5</td>
<td>3/5</td>
</tr>
</tbody>
</table>

Emission

$\{\text{they, can, fish}\}$
Transitions in POS Tagging

Fed raises interest rates 0.5 percent

- $P(y_1 = \text{NNP})$ likely because start of sentence
- $P(y_2 = \text{VBZ}|y_1 = \text{NNP})$ likely because verb often follows noun
- $P(y_3 = \text{NN}|y_2 = \text{VBZ})$: direct object can follow verb

- How are these probabilities learned?
Training HMMs

- Transitions
  - Count up all pairs \((y_i, y_{i+1})\) in the training data
  - Count up occurrences of what tag \(T\) can transition to
  - Normalize to get a distribution for \(P(\text{next tag} | T)\)
  - Need to *smooth* this distribution, won’t discuss here

- Emissions: similar count + normalize scheme, but trickier smoothing!

- You can write down the log likelihood and it is exactly optimized by this count + normalize scheme, so no need for SGD!
Inference: Viterbi Algorithm
Inference in HMMs

- Input $x = (x_1, \ldots, x_n)$   Output $y = (y_1, \ldots, y_n)$

$$P(y, x) = P(y_1) \prod_{i=2}^{n} P(y_i | y_{i-1}) \prod_{i=1}^{n} P(x_i | y_i)$$

- Inference problem: $\text{argmax}_y P(y|x) = \text{argmax}_y \frac{P(y, x)}{P(x)}$

- Exponentially many possible $y$ here!

- Solution: dynamic programming (possible because of Markov structure!)
\[
P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots, y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i)
\]

\[
\max_{y_1, y_2, \cdots, y_n} P(y_n|y_{n-1}) P(x_n|y_n) \cdots P(y_2|y_1) P(x_2|y_2) P(y_1) P(x_1|y_1)
\]

- Transition probabilities
- Emission probabilities
- Initial probability

Slide credit: Vivek Srikumar
\[ P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i) \]

\[
\begin{align*}
\max_{y_1, y_2, \cdots, y_n} & \quad P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1) \\
= \max_{y_2, \cdots, y_n} & \quad P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)
\end{align*}
\]

The only terms that depend on \( y_1 \)

---

slide credit: Vivek Srikumar
\[ P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i) \]

\[
\begin{align*}
\max_{y_1, y_2, \cdots, y_n} & \quad P(y_n|y_{n-1})P(x_n|y_n) \cdots P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1) \\
& = \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1) \\
& = \max_{y_2, \cdots, y_n} P(y_n|y_{n-1})P(x_n|y_n) \cdots \max_{y_1} P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)
\end{align*}
\]

Abstract away the score for all decisions till here into \text{score}

\[
\text{score}_1(s) = P(s)P(x_1|s)
\]

\> Best (partial) score for a sequence ending in state \( s \)
\[ P(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1}|y_i) \prod_{i=1}^{n} P(x_i|y_i) \]

\[
\max_{y_1, y_2, \ldots, y_n} \ P(y_n|y_{n-1})P(x_n|y_n)\cdots
\max_{y_1}P(y_2|y_1)P(x_2|y_2)P(y_1)P(x_1|y_1)
\]

\[
= \max_{y_2, \ldots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_1}P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)
\]

\[
= \max_{y_2, \ldots, y_n} P(y_n|y_{n-1})P(x_n|y_n)\cdots \max_{y_2}P(y_3|y_2)P(x_3|y_3)\max_{y_1}P(y_2|y_1)P(x_2|y_2)\text{score}_1(y_1)
\]

Only terms that depend on \(y_2\)

slide credit: Vivek Srikumar
\[ P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1} | y_i) \prod_{i=1}^{n} P(x_i | y_i) \]

\[
\max_{y_1, y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1)
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1)
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_3 | y_2) P(x_3 | y_3) \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) \text{score}_1(y_1)
= \max_{y_3, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_2} P(y_3 | y_2) P(x_3 | y_3) \text{score}_2(y_2)
\]

\[ \text{score}_i(s) = \max_{y_{i-1}} P(s | y_{i-1}) P(x_i | s) \text{score}_{i-1}(y_{i-1}) \]

Abstract away the score for all decisions till here into score
Viterbi Algorithm

- “Think about” all possible immediate prior state values. Everything before that has already been accounted for by earlier stages.
\[ P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1} | y_i) \prod_{i=1}^{n} P(x_i | y_i) \]

\[
\max_{y_1, y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1)
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1)
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_3 | y_2) P(x_3 | y_3) \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) \text{score}_1(y_1)
= \max_{y_3, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_2} P(y_3 | y_2) P(x_3 | y_3) \text{score}_2(y_2)
= \max_{y_3, \cdots, y_n} \text{score}_n(y_n)
\]

Abstract away the score for all decisions till here into \text{score}
\[ P(x_1, x_2, \cdots, x_n, y_1, y_2, \cdots y_n) = P(y_1) \prod_{i=1}^{n-1} P(y_{i+1} | y_i) \prod_{i=1}^{n} P(x_i | y_i) \]

\[
\begin{align*}
\max_{y_1, y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1) \\
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) P(y_1) P(x_1 | y_1) \\
= \max_{y_2, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) \text{score}_1(y_1) \\
= \max_{y_3, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_2} P(y_3 | y_2) P(x_3 | y_3) \max_{y_1} P(y_2 | y_1) P(x_2 | y_2) \text{score}_1(y_1) \\
= \max_{y_3, \cdots, y_n} P(y_n | y_{n-1}) P(x_n | y_n) \cdots \max_{y_2} P(y_3 | y_2) P(x_3 | y_3) \text{score}_2(y_2) \\
\vdots \\
= \max_{y_n} \text{score}_n(y_n)
\end{align*}
\]

\[
\text{score}_1(s) = P(s)P(x_1 | s)
\]

\[
\text{score}_i(s) = \max_{y_{i-1}} P(s | y_{i-1}) P(x_i | s) \text{score}_{i-1}(y_{i-1})
\]
1. **Initial**: For each state $s$, calculate

$$
score_1(s) = P(s)P(x_1|s) = \pi_s B_{x_1,s}
$$

2. **Recurrence**: For $i = 2$ to $n$, for every state $s$, calculate

$$
score_i(s) = \max_{y_{i-1}} P(s|y_{i-1})P(x_i|s)score_{i-1}(y_{i-1})
= \max_{y_{i-1}} A_{y_{i-1},s} B_{s,x_i} score_{i-1}(y_{i-1})
$$

3. **Final state**: calculate

$$
\max_{y} P(y, x|\pi, A, B) = \max_{s} score_n(s)
$$

This only calculates the max. To get final answer (argmax),

- keep track of which state corresponds to the max at each step
- build the answer using these back pointers

slide credit: Vivek Srikumar
POS Taggers
HMM POS Tagging

- Penn Treebank English POS tagging: 44 tags
- Baseline: assign each word its most frequent tag: ~90% accuracy
- Trigram HMM (states are *pairs* of tags): ~95% accuracy / 55% on words not seen in train
- TnT tagger (Brants 1998, tuned HMM): 96.2% acc / 86.0% on unks
- CRF tagger (Toutanova + Manning 2000): 96.9% / 87.0%
- State-of-the-art (BiLSTM-CRFs, BERT): 97.5% / 89%+

Slide credit: Dan Klein
## Errors

<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>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>
<td>488</td>
</tr>
<tr>
<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>
<td>525</td>
</tr>
<tr>
<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>
<td>427</td>
</tr>
<tr>
<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>
<td>142</td>
</tr>
<tr>
<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>
<td>295</td>
</tr>
<tr>
<td>0</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>104</td>
</tr>
<tr>
<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>
<td>323</td>
</tr>
<tr>
<td>17</td>
<td>64</td>
<td>9</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>4</td>
<td>7</td>
<td>85</td>
<td>189</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>143</td>
<td>2</td>
<td>166</td>
<td></td>
</tr>
<tr>
<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>
<td>221</td>
</tr>
<tr>
<td>5</td>
<td>34</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>49</td>
<td>6</td>
<td>3</td>
<td>0</td>
<td>104</td>
</tr>
<tr>
<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>
<td>3651</td>
</tr>
</tbody>
</table>

**Slide credit:** Dan Klein / Toutanova + Manning (2000)
Remaining Errors

- Lexicon gap (word not seen with that tag in training) 4.5%
- Unknown word: 4.5%
- Could get right: 16% (many of these involve parsing!)
- Difficult linguistics: 20%

  VBD / VBP? (past or present?)
  
  They \textit{set} up absurd situations, detached from reality

- Underspecified / unclear, gold standard inconsistent / wrong: 58%

  adjective or verbal participle? JJ / VBN?
  
  a $10$ million fourth-quarter charge against \textit{discontinued} operations

Manning 2011 “Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?”
Other Languages

<table>
<thead>
<tr>
<th>Language</th>
<th>CRF+</th>
<th>CRF</th>
<th>BTS</th>
<th>BTS*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bulgarian</td>
<td>97.97</td>
<td>97.00</td>
<td>97.84</td>
<td>97.02</td>
</tr>
<tr>
<td>Czech</td>
<td>98.38</td>
<td>98.00</td>
<td>98.50</td>
<td>98.44</td>
</tr>
<tr>
<td>Danish</td>
<td>95.93</td>
<td>95.06</td>
<td>95.52</td>
<td>92.45</td>
</tr>
<tr>
<td>German</td>
<td>93.08</td>
<td>91.99</td>
<td>92.87</td>
<td>92.34</td>
</tr>
<tr>
<td>Greek</td>
<td>97.72</td>
<td>97.21</td>
<td>97.39</td>
<td>96.64</td>
</tr>
<tr>
<td>English</td>
<td>95.11</td>
<td>94.51</td>
<td>93.87</td>
<td>94.00</td>
</tr>
<tr>
<td>Spanish</td>
<td>96.08</td>
<td>95.03</td>
<td>95.80</td>
<td>95.26</td>
</tr>
<tr>
<td>Farsi</td>
<td>96.59</td>
<td>96.25</td>
<td>96.82</td>
<td>96.76</td>
</tr>
<tr>
<td>Finnish</td>
<td>94.34</td>
<td>92.82</td>
<td>95.48</td>
<td>96.05</td>
</tr>
<tr>
<td>French</td>
<td>96.00</td>
<td>95.93</td>
<td>95.75</td>
<td>95.17</td>
</tr>
<tr>
<td>Indonesian</td>
<td>92.84</td>
<td>92.71</td>
<td>92.85</td>
<td>91.03</td>
</tr>
<tr>
<td>Italian</td>
<td>97.70</td>
<td>97.61</td>
<td>97.56</td>
<td>97.40</td>
</tr>
<tr>
<td>Swedish</td>
<td>96.81</td>
<td>96.15</td>
<td>95.57</td>
<td>93.17</td>
</tr>
<tr>
<td>AVERAGE</td>
<td>96.04</td>
<td>95.41</td>
<td>95.85</td>
<td>95.06</td>
</tr>
</tbody>
</table>

- Universal POS tagset (~12 tags), cross-lingual model works as well as tuned CRF using external resources

Gillick et al. 2016

```
Óscar Romero was born in El Salvador.

SEGMENT

Ó

SPANS

[S0, L13, PER]  [S26, L11, LOC]
```
Named Entity Recognition

- BIO tagset: begin, inside, outside
- Sequence of tags — should we use an HMM?
- Why might an HMM not do so well here?
  - Lots of O’s
  - Insufficient features/capacity with multinomials (especially for unks)
HMMs Pros and Cons

- Big advantage: transitions, scoring pairs of adjacent y’s

- Big downside: not able to incorporate useful word context information

- Solution: switch from generative to discriminative model (conditional random fields) so we can condition on the *entire input*.

- Conditional random fields: logistic regression + features on pairs of y’s
Conditional Random Fields
Conditional Random Fields

- Flexible discriminative model for tagging tasks that can use arbitrary features of the input. Similar to logistic regression, but **structured**

  Barack Obama will travel to Hangzhou today for the G20 meeting.

  Curr_word=Barack & Label=B-PER
  Next_word=Obama & Label=B-PER
  Curr_word_starts_with_capital=True & Label=B-PER
  Posn_in_sentence=1st & Label=B-PER
  Label=B-PER & Next-Label = I-PER
  ...

  B-PER  I-PER
Tagging with Logistic Regression

- Logistic regression over each tag individually:

\[ P(y_i = y|x, i) = \frac{\exp(w^T f(y, i, x))}{\sum_{y' \in \mathcal{Y}} \exp(w^T f(y', i, x))} \]

“different features” approach to features for a single tag

Probability of the \(i\)th word getting assigned tag \(y\) (B-PER, etc.)
Tagging with Logistic Regression

- Logistic regression over each tag individually:
  \[ P(y_i = y | x, i) = \frac{\exp(w^\top f(y, i, x))}{\sum_{y' \in Y} \exp(w^\top f(y', i, x))} \]

- Over all tags:
  \[ P(y = \tilde{y} | x) = \prod_{i=1}^{n} P(y_i = \tilde{y}_i | x, i) = \frac{1}{Z} \exp \left( \sum_{i=1}^{n} w^\top f(\tilde{y}_i, i, x) \right) \]

- Score of a prediction: sum of weights dot features over each individual predicted tag (this is a simple CRF but not the general form)

- Set \( Z \) equal to the product of denominators

- Conditional model: \( x \) is observed, unlike in HMMs
Example: “Emission Features” $f_e$

B-PER  I-PER  O  O
Barack Obama will travel

$\text{feats} = f_e(\text{B-PER}, i=1, x) + f_e(\text{I-PER}, i=2, x) + f_e(\text{O}, i=3, x) + f_e(\text{O}, i=4, x)$

[CurrWord=Obama & label=I-PER, PrevWord=Barack & label=I-PER, CurrWordIsCapitalized & label=I-PER, ...]

B-PER  B-PER  O  O
Barack Obama will travel

$\text{feats} = f_e(\text{B-PER}, i=1, x) + f_e(\text{B-PER}, i=2, x) + f_e(\text{O}, i=3, x) + f_e(\text{O}, i=4, x)$
Adding Structure

\[
P(y = \tilde{y} | x) = \frac{1}{Z} \exp \left( \sum_{i=1}^{n} w^\top f(\tilde{y}_i, i, x) \right)
\]

- We want to be able to learn that some tags don’t follow other tags — want to have features on tag pairs

\[
P(y = \tilde{y} | x) = \frac{1}{Z} \exp \left( \sum_{i=1}^{n} w^\top f_e(\tilde{y}_i, i, x) + \sum_{i=2}^{n} w^\top f_t(\tilde{y}_{i-1}, \tilde{y}_i, i, x) \right)
\]

- Score: sum of weights dot \(f_e\) features over each predicted tag ("emissions") plus sum of weights dot \(f_t\) features over tag pairs ("transitions")

- This is a sequential CRF
Example

B-PER  I-PER  O  O
Barack Obama will travel

feats = $f_e(B\text{-PER}, i=1, x) + f_e(I\text{-PER}, i=2, x) + f_e(O, i=3, x) + f_e(O, i=4, x)$
$+ f_t(B\text{-PER}, I\text{-PER}, i=1, x) + f_t(I\text{-PER}, O, i=2, x) + f_t(O, O, i=3, x)$

B-PER  B-PER  O  O
Barack Obama will travel

feats = $f_e(B\text{-PER}, i=1, x) + f_e(B\text{-PER}, i=2, x) + f_e(O, i=3, x) + f_e(O, i=4, x)$
$+ f_t(B\text{-PER}, B\text{-PER}, i=1, x) + f_t(B\text{-PER}, O, i=2, x) + f_t(O, O, i=3, x)$

- Obama can start a new named entity (emission feats look okay), but we’re not likely to have two PER entities in a row (transition feats)
Sequential CRFs

\[ P(y = \tilde{y} | x) = \frac{1}{Z} \exp \left( \sum_{i=1}^{n} w^\top f_e(\tilde{y}_i, i, x) + \sum_{i=2}^{n} w^\top f_t(\tilde{y}_{i-1}, \tilde{y}_i, i, x) \right) \]

- **Critical property:** this structure is allows us to use dynamic programming (Viterbi) to sum or max over all sequences

- **Inference:** use Viterbi, just replace probabilities with exponentiated weights * features

- **Learning:** need another dynamic program (forward-backward) to compute gradients
Can generalize CRFs to work with neural networks (including BERT): “neural CRFs” for tagging (Lample et al., 2016), parsing (Durrett and Klein, 2015; Dozat and Manning, 2016)

Why aren’t CRFs used more today?

- We don’t often need to score transitions: If you have hard constraints (e.g., cannot follow B-PER with I-ORG), you can simply integrate these into inference. Train BERT to predict each label individually, then use Viterbi to get a coherent sequence.

- ChatGPT and other such systems are decent at learning structural constraints — so bigger models also learn most of the constraints you really want
Takeaways

› POS and NER are two ways of capturing sequential structures

› POS: syntax, each word has a tag

› NER: spans, but we can turn them into tags with BIO

› Can handle these with generative or discriminative models, but CRFs are most typically used (although these days you can also just ask ChatGPT...)

› Next time: move from sequences to trees