Administrivia

- Kyunghyun Cho (NYU) talk Friday 11am GDC 6.302
- Project 3 due today!
- Final project out today!
  - Proposal due in 1 week
  - Project presentations December 5/7 (timeslots to be assigned when proposals are turned in)
  - Final project due December 15 (no slip days!)
Project Proposals

- ~1 page
  - Define a problem, give context of related work (at least 3-4 relevant papers)
  - Propose a direction that you think is feasible and outline steps to get there, including what dataset you’ll use

- Okay to change directions after the proposal is submitted, but run it by me if it’s a big change
This Lecture

- Neural CRFs
- Tagging / NER
- Parsing
Neural CRF Basics
NER Revisited

**Features in CRFs:**
- I[tag=B-LOC & curr_word=Hangzhou],
- I[tag=B-LOC & prev_word=to], I[tag=B-LOC & curr_prefix=Han]

**Linear model over features**

**Downsides:**
- Lexical features mean that words need to be seen in the training data
- Can only use limited context windows
- Linear model can’t capture feature conjunctions effectively

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

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

‣ What are the strengths and weaknesses of this model compared to CRFs?

‣ Encoder-decoder (MT-like model)
LSTMs for NER

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

- **Transducer (LM-like model)**
- **What are the strengths and weaknesses of this model compared to CRFs?**
LSTMs for NER

<table>
<thead>
<tr>
<th>B-PER</th>
<th>I-PER</th>
<th>O</th>
<th>O</th>
<th>O</th>
<th>B-LOC</th>
<th>O</th>
<th>O</th>
<th>O</th>
<th>B-ORG</th>
<th>O</th>
<th>O</th>
</tr>
</thead>
</table>

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

- **Bidirectional transducer model**
- What are the strengths and weaknesses of this model compared to CRFs?
Barack Obama will travel to Hangzhou today for the G20 meeting.

Neural CRFs: bidirectional LSTMs (or some NN) compute emission potentials, capture structural constraints in transition potentials.
Neural CRFs

\[ P(y|x) = \frac{1}{Z} \prod_{i=2}^{n} \exp(\phi_t(y_{i-1}, y_i)) \prod_{i=1}^{n} \exp(\phi_e(y_i, i, x)) \]

- Conventional: \( \phi_e(y_i, i, x) = w^\top f_e(y_i, i, x) \)
- Neural: \( \phi_e(y, i, x) = W f(i, x) \) \( f/\phi \) are vectors, \( \text{len}(\phi) = \text{num labels} \)
- \( f(i, x) \) could be the output of a feedforward neural network looking at the words around position \( i \), or the \( i \)th output of an LSTM, ...
- Neural network computes unnormalized potentials that are consumed and “normalized” by a structured model
- Inference: compute \( f \), use Viterbi (or beam)
Computing Gradients

\[
P(y|\mathbf{x}) = \frac{1}{Z} \prod_{i=2}^{n} \exp(\phi_t(y_{i-1}, y_i)) \prod_{i=1}^{n} \exp(\phi_e(y_i, i, \mathbf{x}))
\]

- Conventional: \( \phi_e(y_i, i, \mathbf{x}) = w^\top f_e(y_i, i, \mathbf{x}) \)

- Neural: \( \phi_e(y, i, \mathbf{x}) = W f(i, \mathbf{x}) \)

\[
\frac{\partial \mathcal{L}}{\partial \phi_{e,i}} = -P(y_i = s|\mathbf{x}) + I[s \text{ is gold}] \quad \text{“error signal”, compute with F-B chain rule say to multiply together, gives our update}
\]

- For linear model: \( \frac{\partial \phi_{e,i}}{w_i} = f_{e,i}(y_i, i, \mathbf{x}) \)

- For neural model: compute gradient of \( \phi \) w.r.t. parameters of neural net
Neural CRFs

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

1) Compute $f(x)$
2) Run forward-backward
3) Compute error signal
4) Backprop (no knowledge of sequential structure required)
Barack Obama will travel to Hangzhou today for the G20 meeting.

\[
\phi_e = W g(V f(x, i))
\]
\[
f(x, i) = [\text{emb}(x_{i-1}), \text{emb}(x_i), \text{emb}(x_{i+1})]
\]

\(\text{to Hangzhou today}\)
Neural CRFs

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

- Neural CRFs: bidirectional LSTMs compute emission potentials, also transition potentials (usually based on sparse features)
Barack Obama will travel to Hangzhou today for the G20 meeting.

How does this compare to neural CRF?
“NLP (Almost) From Scratch”

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHUNK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+WLL</td>
<td>96.31</td>
<td>89.13</td>
<td>79.53</td>
<td>55.40</td>
</tr>
<tr>
<td>NN+SLL</td>
<td>96.37</td>
<td>90.33</td>
<td>81.47</td>
<td>70.99</td>
</tr>
<tr>
<td>NN+WLL+LM1</td>
<td>97.05</td>
<td>91.91</td>
<td>85.68</td>
<td>58.18</td>
</tr>
<tr>
<td>NN+SLL+LM1</td>
<td>97.10</td>
<td>93.65</td>
<td>87.58</td>
<td>73.84</td>
</tr>
<tr>
<td>NN+WLL+LM2</td>
<td>97.14</td>
<td>92.04</td>
<td>86.96</td>
<td>58.34</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>74.15</td>
</tr>
</tbody>
</table>

- WLL: independent classification; SLL: neural CRF
- LM1/LM2: pretrained word embeddings from a language model over large corpora

How do we use a tagger for SRL?

- Tagging problem with respect to a particular verb

- Can’t do this with feedforward networks efficiently, arguments are too far from the verb to use fixed context window sizes

Figure from He et al. (2017)
CNN Neural CRFs

- Append to each word vector an *embedding of the relative position of that word*
- Convolution over the sentence produces a position-dependent representation
- Use this for SRL: the verb (predicate) is at position 0, CNN looks at the whole sentence “relative” to the verb

*expected* to *quicken* *a* *bit*
### CNN NCRFs vs. FFNN NCRFs

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHUNK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>Window Approach</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>–</td>
</tr>
<tr>
<td>Sentence Approach</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.12</td>
<td>93.37</td>
<td>88.78</td>
<td>74.15</td>
</tr>
</tbody>
</table>

- Sentence approach (CNNs) is comparable to window approach (FFNNs) except for SRL where they claim it works much better.
How “from scratch” was this?

<table>
<thead>
<tr>
<th>Approach</th>
<th>POS (PWA)</th>
<th>CHUNK (F1)</th>
<th>NER (F1)</th>
<th>SRL (F1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark Systems</td>
<td>97.24</td>
<td>94.29</td>
<td>89.31</td>
<td>77.92</td>
</tr>
<tr>
<td>NN+WLL</td>
<td>96.31</td>
<td>89.13</td>
<td>79.53</td>
<td>55.40</td>
</tr>
<tr>
<td>NN+SLL</td>
<td>96.37</td>
<td>90.33</td>
<td>81.47</td>
<td>70.99</td>
</tr>
<tr>
<td>NN+WLL+LM1</td>
<td>97.05</td>
<td>91.91</td>
<td>85.68</td>
<td>58.18</td>
</tr>
<tr>
<td>NN+SLL+LM1</td>
<td>97.10</td>
<td>93.65</td>
<td>87.58</td>
<td>73.84</td>
</tr>
<tr>
<td>NN+WLL+LM2</td>
<td>97.14</td>
<td>92.04</td>
<td>86.96</td>
<td>58.34</td>
</tr>
<tr>
<td>NN+SLL+LM2</td>
<td>97.20</td>
<td>93.63</td>
<td>88.67</td>
<td>74.15</td>
</tr>
<tr>
<td>NN+SLL+LM2+Suffix2</td>
<td>97.29</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+Gazetteer</td>
<td>–</td>
<td>–</td>
<td>89.59</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+POS</td>
<td>–</td>
<td>94.32</td>
<td>88.67</td>
<td>–</td>
</tr>
<tr>
<td>NN+SLL+LM2+CHUNK</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>74.72</td>
</tr>
</tbody>
</table>

- NN+SLL isn’t great
- LM2: trained for 7 weeks on Wikipedia+Reuters — very expensive!
- Sparse features needed to get best performance on NER+SRL anyway
- No use of sub-word features...

Collobert and Weston 2008, 2011
Neural CRF using character LSTMs to compute word representations

Chiu and Nichols (2015), Lample et al. (2016)
Neural CRFs with LSTMs

- Chiu+Nichols: character CNNs instead of LSTMs
- Lin/Passos/Luo: use external resources like Wikipedia
- LSTM-CRF captures the important aspects of NER: word context (LSTM), sub-word features (character LSTMs), outside knowledge (word embeddings)

<table>
<thead>
<tr>
<th>Model</th>
<th>$F_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collobert et al. (2011)*</td>
<td>89.59</td>
</tr>
<tr>
<td>Lin and Wu (2009)</td>
<td>83.78</td>
</tr>
<tr>
<td>Lin and Wu (2009)*</td>
<td>90.90</td>
</tr>
<tr>
<td>Huang et al. (2015)*</td>
<td>90.10</td>
</tr>
<tr>
<td>Passos et al. (2014)</td>
<td>90.05</td>
</tr>
<tr>
<td>Passos et al. (2014)*</td>
<td>90.90</td>
</tr>
<tr>
<td>Luo et al. (2015)* + gaz</td>
<td>89.9</td>
</tr>
<tr>
<td>Luo et al. (2015)* + gaz + linking</td>
<td><strong>91.2</strong></td>
</tr>
<tr>
<td>Chiu and Nichols (2015)</td>
<td>90.69</td>
</tr>
<tr>
<td>Chiu and Nichols (2015)*</td>
<td>90.77</td>
</tr>
<tr>
<td>LSTM-CRF (no char)</td>
<td>90.20</td>
</tr>
<tr>
<td>LSTM-CRF</td>
<td><strong>90.94</strong></td>
</tr>
</tbody>
</table>

Chiu and Nichols (2015), Lample et al. (2016)
Neural CRFs for Parsing
He wrote a long report on Mars.

He wrote a long report on Mars.

My report—on Mars.

report—on Mars.

wrote—on Mars.
Discrete Parsing

$$\text{score} \left( \begin{array}{c} \text{NP} \\ \text{NP} \\ \text{PP} \end{array} \right) = w^T f \left( \begin{array}{c} \text{NP} \\ \text{NP} \\ \text{PP} \\ \text{NP} \\ \text{NP} \\ \text{PP} \end{array} \right)$$

He wrote a long report on Mars.

Drawbacks

- Need to learn each word’s properties individually
- Hard to learn feature conjunctions (report on X)

Taskar et al. (2004)  
Hall, Durrett, and Klein (ACL 2014)
Continuous-State Grammars

He wrote a long report on Mars.

Powerful nonlinear featurization, but inference is intractable

Socher et al. (2013)
Joint Discrete and Continuous Parsing

\[
\text{score}\left( \begin{array}{c}
  \text{NP} \\
  2 \\
  \text{NP} \\
  5 \\
  \text{PP} \\
  7 \\
\end{array} \right) = w^T f\left( \begin{array}{c}
  \text{NP} \\
  2 \\
  \text{NP} \\
  5 \\
  \text{PP} \\
  7 \\
\end{array} \right)
\]

He wrote a long report on Mars

\[
+ s^T \left( \begin{array}{c}
  X \\
  2 \\
  X \\
  5 \\
  X \\
  7 \\
\end{array} \right) W \ell\left( \begin{array}{c}
  \text{NP} \\
  \text{NP} \\
  \text{PP} \\
\end{array} \right)
\]
He wrote a long report on Mars

$$\text{score} \left( \begin{array}{c} \text{NP} \\ 2 \\ \text{NP} \\ 5 \\ \text{PP} \\ 7 \end{array} \right)$$

$$= w^\top f \left( \begin{array}{c} \text{NP} \\ 2 \\ \text{NP} \\ 5 \\ \text{PP} \\ 7 \end{array} \right) + s^\top \left( \begin{array}{c} X \\ 2 \\ X \\ 5 \\ X \\ 7 \end{array} \right) W \ell \left( \begin{array}{c} \text{NP} \\ \text{NP} \\ \text{PP} \end{array} \right)$$

Durrett and Klein (ACL 2015)
He wrote a long report on Mars.

Durrett and Klein (ACL 2015)
Joint Discrete and Continuous Parsing

- Chart remains discrete!

Parsing a sentence:
- Feedforward pass on nets
- Discrete feature computation
- Run CKY dynamic program

Durrett and Klein (ACL 2015)
Joint Modeling Helps!

Approx human performance

95

Penn Treebank Dev set F_1

90.8 Discrete
Hall, Durrett, Klein 2014

91.0 Continuous
Durrett, Klein 2015

92.0 Joint
Durrett, Klein 2015
Comparison to Neural Parsers

Approx human performance: 95

Penn Treebank Test set F₁

- CVG: 90.4 (Socher+ 2013)
- LSTM: 88.3 (Vinyals+ 2015)
- LSTM ensemble: 90.5 (Vinyals+ 2015)
- Joint: 91.1 (Durrett, Klein 2015)

Approx human performance: 95
Results: 8 languages

- SPMRL Treebanks
  - Test set $F_1$
  - Berkeley Petrov+ 2006: 81.2
  - FM Fernanades-Gonzalez and Martins 2015: 84.2
  - Joint Durrett, Klein 2015: 85.7
Score each head-child pair in a dependency parse, use Eisner’s algorithm or MST to assemble a parse.

Feedforward neural network approach: use features on head and modifier.

Pei et al. (2015), Kiperwasser and Goldberg (2016), Dozat and Manning (2017)
Dependency Parsing

- Biaffine approach: condense each head and modifier separately, compute score $h^TUm$

MLP: $h^{(arc-dep)}_i, h^{(arc-head)}_i$

BiLSTM: $r_i$

Embeddings: $x_i$
### Results

<table>
<thead>
<tr>
<th>Type</th>
<th>Model</th>
<th>English PTB-SD 3.3.0</th>
<th>Chinese PTB 5.1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>UAS</td>
<td>LAS</td>
</tr>
<tr>
<td></td>
<td>Ballesteros et al. (2016)</td>
<td>93.56</td>
<td>91.42</td>
</tr>
<tr>
<td>Transition</td>
<td>Andor et al. (2016)</td>
<td>94.61</td>
<td>92.79</td>
</tr>
<tr>
<td>Transition</td>
<td>Kuncoro et al. (2016)</td>
<td>95.8</td>
<td>94.6</td>
</tr>
<tr>
<td>Transition</td>
<td>Kiperwasser &amp; Goldberg (2016)</td>
<td>93.9</td>
<td>91.9</td>
</tr>
<tr>
<td>Graph</td>
<td>Cheng et al. (2016)</td>
<td>94.10</td>
<td>91.49</td>
</tr>
<tr>
<td>Graph</td>
<td>Hashimoto et al. (2016)</td>
<td>94.67</td>
<td>92.90</td>
</tr>
<tr>
<td>Graph</td>
<td>Deep Biaffine</td>
<td>95.74</td>
<td>94.08</td>
</tr>
</tbody>
</table>

- Biaffine approach works well (other neural CRFs are also strong)

Dozat and Manning (2017)
Neural CRFs

- State-of-the-art for:
  - POS
  - NER without extra data (Lample et al.)
  - Dependency parsing (Dozat and Manning)
  - Semantic Role Labeling (He et al.)

- Why do they work so well?
  - Word-level LSTMs compute features based on the word + context
  - Character LSTMs/CNNs extract features per word
  - Pretrained embeddings capture external semantic information
  - CRF handles structural aspects of the problem
Takeaways

- Any structured model / dynamic program + any neural network to compute potentials = neural CRF

- Can incorporate transition potentials or other scores over the structure like grammar rules

- State-of-the-art for many text analysis tasks