Stanford’s Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task

Timothy Dozat, Peng Qi, Christopher D. Manning

Stanford University

August 6, 2017
Overview

1. Parser
2. UPOS/XPOS tagger
3. Character-level embedding model
4. Results
5. Noteworthy hyperparameters
Almost everything builds on this structure (cf. Dozat and Manning (2017)):
<table>
<thead>
<tr>
<th>Overview</th>
<th>Parser</th>
<th>Tagger</th>
<th>Character Model</th>
<th>Results</th>
<th>Hyperparameters</th>
<th>References</th>
</tr>
</thead>
</table>

**Parser**

Timothy Dozat, Peng Qi, Christopher D. Manning
Stanford’s Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task
Unlabeled parser: LSTM

- Bidirectional LSTM over word/tag embeddings (more on embeddings later)
- Two separate FC ReLU layers
  - One representing each token as a dependent trying to find (attend to) its head
  - One representing each token as a head trying to find (be attended to by) its dependents

\[ h_i^{(arc-head)} \]
\[ h_i^{(arc-dep)} \]

FC/ReLU

BiLSTM

Timothy Dozat, Peng Qi, Christopher D. Manning
Stanford University
Stanford’s Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task
Unlabeled parser: Self-attention

- Biaffine self-attention layer to score possible heads for each dependent

\[ s_i^{(arc)} = H^{(arc-head)} \cdot W \oplus b \cdot h_i^{(arc-dep)} \oplus 1 \]

- Train with cross-entropy
- Apply a spanning tree algorithm at inference time

Note: This is just an affine layer with a linear transformation!
Labeler: LSTM

- Take the topmost BiLSTM vectors used for the unlabeled parser.
- Two more separate FC ReLU layers:
  - One representing each token as a dependent trying to determine its label.
  - One representing each token as a head trying to determine its dependents’ labels.
Labeler: Classifier

- Biaffine layer to score possible relations for each best-head/dependent pair

\[
\begin{align*}
s_i^{(rel)} \quad h_{yi}^{(rel-head)} \oplus 1 \\
\quad \top = \quad \ast \quad U \quad \ast \\
\quad \top \quad = \quad \ast \\
\end{align*}
\]

- Train with softmax cross-entropy, added to the loss of the unlabeled parser

Note: this is just a linear model with interaction effects!

\[
\text{label.scores} \sim \text{head.state} \times \text{dep.state}
\]

Timothy Dozat, Peng Qi, Christopher D. Manning
Stanford University
Stanford’s Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task
Tagger
Tagger: Motivation

- **Problem**: Dozat and Manning’s (2017) parser had lower label accuracy than we wanted
- **Idea**: Better POS tag quality might improve label score
- **Question**: Will improved POS tag accuracy result in better parsers?
Tagger: LSTM

- BiLSTM (distinct from parser BiLSTM!) over word embeddings
- Two separate FC ReLU layers:
  - One for UPOS tags
  - One for XPOS tags

\[
\begin{align*}
  h_i^{(upos)} & \quad h_i^{(xpos)} \\
  \text{FC/ReLU} & \\
  \text{BiLSTM} &
\end{align*}
\]
Tagger: Classifiers

- Affine layers to score possible tags for each word

\[
\begin{align*}
    s_i^{(pos)} &= W \oplus b \\
    h_i^{(pos)} &= 1 \\
    h_i^{(pos)} &= W \cdot s_i^{(pos)} + b
\end{align*}
\]

- Train jointly by adding together softmax cross-entropy

- When using in the main parser, add UPOS and XPOS embeddings together (eltwise)
Tagger: Experiment

- Systems with our tagger outperformed systems with baseline tagger (Straka et al., 2015) \((p < .05)\) or no tagger \((p < .05)\)
- Parser performance correlated with tagger performance (ours vs. baseline) \((p < .05)\)
Character Model
Character model: Motivation

- **Problem**: Many shared task languages have complex morphology
  - Grammatical functions indicated more by word form than relative location
  - Rare words with highly predictive suffixes won’t be attested in the frequent word embedding matrix
  - Extreme sparsity may yield low-quality pretrained embeddings
- **Idea**: Compose word embeddings orthographically with a character-based embedding model
- **Question**: Does this improve accuracy on inflectionally rich languages?
Character model: LSTM

- Unidirectional LSTM over character embeddings
- Concatenate two sources of information:
  - Linear attention over top hidden states (Cao and Rei, 2016)
  - Final cell state (Ballesteros et al., 2015)
Character model: Embedding

- Linearly transform to the desired size

Character model: Embedding

- When using in the parser/tagger, add with pretrained and frequent-token embeddings (eltwise)
Character model: Experiment

- Systems trained with a character model outperformed models trained without ($p < .05$)
- Improvement correlated with morphological complexity ($p < .05$)

Effect of Morphological Complexity on Parser

Effect of Morphological Complexity on Tagger

Timothy Dozat, Peng Qi, Christopher D. Manning
Stanford's Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task
Results
### Results

<table>
<thead>
<tr>
<th>Treebanks</th>
<th>UPOS</th>
<th>XPOS</th>
<th>UAS</th>
<th>LAS</th>
<th>CLAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>All treebanks</td>
<td>93.09</td>
<td>82.27</td>
<td>81.30</td>
<td>76.30</td>
<td>72.57</td>
</tr>
<tr>
<td>Large treebanks</td>
<td>95.58</td>
<td>94.56</td>
<td>85.16</td>
<td>81.77</td>
<td>78.40</td>
</tr>
<tr>
<td>Parallel treebanks</td>
<td>88.25</td>
<td>30.66</td>
<td>80.17</td>
<td>73.73</td>
<td>69.88</td>
</tr>
<tr>
<td>Small treebanks</td>
<td>87.02</td>
<td>82.03</td>
<td>70.19</td>
<td>61.02</td>
<td>54.76</td>
</tr>
<tr>
<td>Surprise treebanks</td>
<td>–</td>
<td>–</td>
<td>54.47</td>
<td>40.57</td>
<td>37.41</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System</th>
<th>UPOS</th>
<th>XPOS</th>
<th>UAS</th>
<th>LAS</th>
<th>CLAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dozat et al.</td>
<td>93.09</td>
<td>82.27</td>
<td>81.30</td>
<td>76.30</td>
<td>72.57</td>
</tr>
<tr>
<td>Björkelund et al.</td>
<td>91.98</td>
<td>64.84</td>
<td>79.90</td>
<td>74.42</td>
<td>70.18</td>
</tr>
<tr>
<td>Yu et al.</td>
<td>91.00</td>
<td>79.93</td>
<td>74.22</td>
<td>68.41</td>
<td>63.24</td>
</tr>
<tr>
<td>Shi et al.</td>
<td>90.88</td>
<td>79.80</td>
<td>80.35</td>
<td>75.00</td>
<td>70.91</td>
</tr>
</tbody>
</table>

*Stanford’s Graph-based Neural Dependency Parser at the CoNLL 2017 Shared Task*
Hyperparameters

(Protips)
Noteworthy hyperparameters

**Dropout**

- Lots of dropout: `keep_prob` is .67 throughout the whole network
- Embedding dropout
  - Drop token/tag embeddings independently
  - When one is dropped, the other is scaled up to compensate
  - When both are dropped, replace with zeros
  - Seems to work better than random vector/UNK replacement
- Same-mask recurrent dropout (Gal and Ghahramani, 2016)
  - Drop input connections *and* recurrent connections
  - Drop the same connections at each recurrent timestep
  - Seems to work better than traditional dropout/zoneout (Krueger et al., 2017)
Noteworthy hyperparameters

Adam

- Adam optimizer (Kingma and Ba, 2015) with $\beta_1 = \beta_2 = .9$
- For embedding matrices, only decay $m$ and $v$ accumulators for tokens that were used in the minibatch
  - I.e. for words that are attested in the minibatch, we apply Adam’s accumulator update rule:
    \[
    m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \\
    v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2
    \]
- But for words that aren’t, we don’t update the accumulators, preventing them from decaying down to zero for uncommon words
- Note: this is not the behavior of most ML toolkits!
Noteworthy hyperparameters

Initialization

- Preference for initializing to zero wherever possible
  - Bias terms
  - Final linear layers (character model, output layers)
  - Word/POS embeddings (other than pretrained)
- Otherwise, we use orthonormal initialization (Saxe et al., 2014)

Recurrent Cells

- LSTMs vastly outperformed GRUs and slightly outperformed coupled input-forget LSTMs (Greff et al., 2016)
- Adding a forget bias hurts performance
Nonprojectivity

- Our system outperforms UDPipe v1.1 (transition-based) by a larger margin on treebanks with many crossing arcs ($p < .05$)
- Stronger correlation for treebanks with more crossing arcs in the test set than in the training set ($p < .05$)
Thanks for listening!


References III


