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

Timothy Dozat Peng Qi Christopher D. Manning

Stanford University

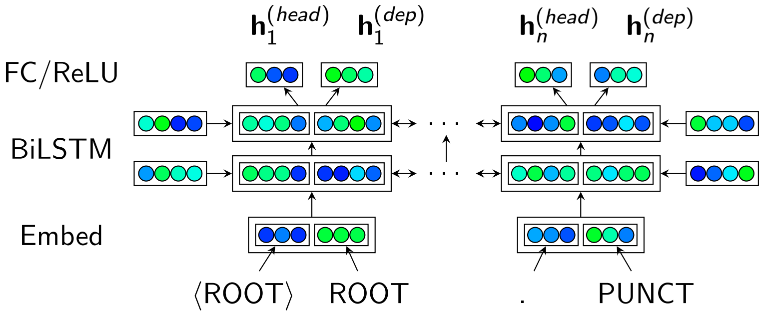
August 6, 2017

Overview

1. Parser
2. UPOS/XPOS tagger
3. Character-level embedding model
4. Results
5. Noteworthy hyperparameters

Overview: Architecture

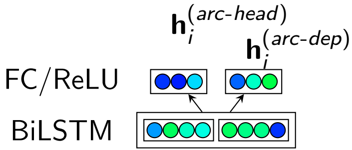
Almost everything builds on this structure (cf. Dozat and Manning (2017)):



Parser

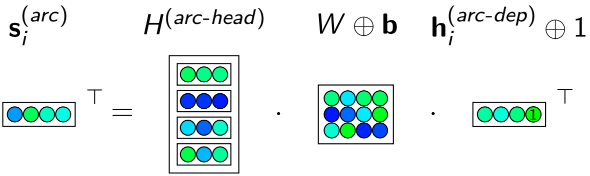
Unlabeled parser: LSTM

- 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



Unlabeled parser: Self-attention

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



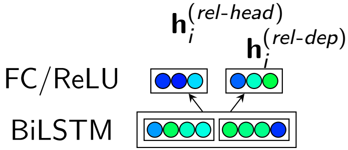
- Train with cross-entropy
- Apply a spanning tree algorithm at inference time

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

$$\mathbf{s}_i = H^{(arc\text{-}head)}(W\mathbf{h}_i^{(arc\text{-}dep)} + \mathbf{b})$$

Labeler: LSTM

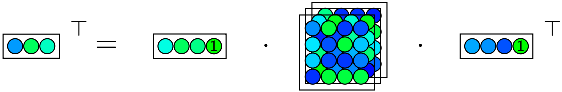
- Take the topmost BiLSTM vectors used for the unlabeled parser
- Two more separate FC ReLU layers:
  - One representing each token as a dependent trying to determine its label
  - One representing each token as a head trying to determine its dependents' labels



Labeler: Classifier

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

$$\mathbf{s}_i^{(rel)\top} = \mathbf{h}_{y_i}^{(rel\text{-}head)} \oplus 1 \cdot \mathbf{U} \cdot \left(\mathbf{h}_i^{(rel\text{-}dep)} \oplus 1\right)^\top$$



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

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

```
label.scores ~ head.state * dep.state
```

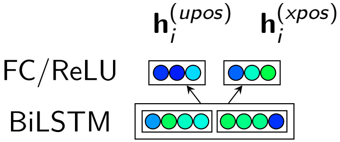
Tagger

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

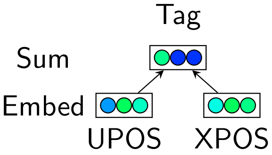


Tagger: Classifiers

- Affine layers to score possible tags for each word

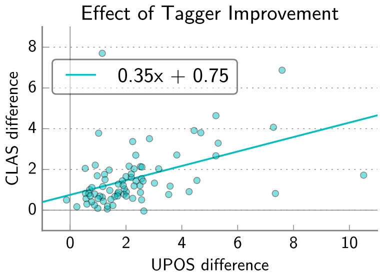
$$ {\mathbf{s}_i^{(pos)}}^\top = (W \oplus \mathbf{b}) \cdot {(\mathbf{h}_i^{(pos)} \oplus 1)}^\top \times 2 $$

- 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) ($p < .05$)



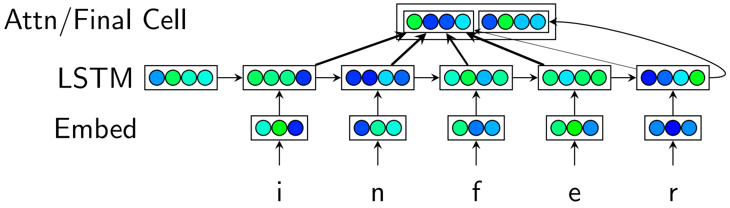
Character Model

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?

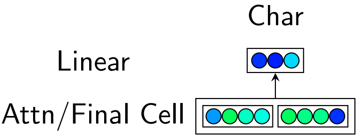
Character model: LSTM

- 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)

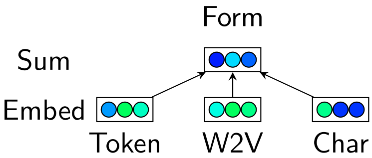


Character model: Embedding

- Linearly transform to the desired size

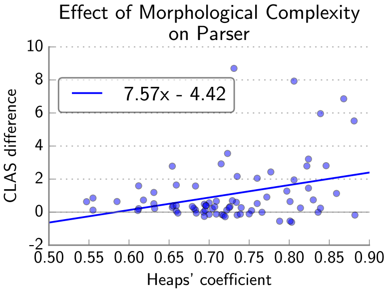


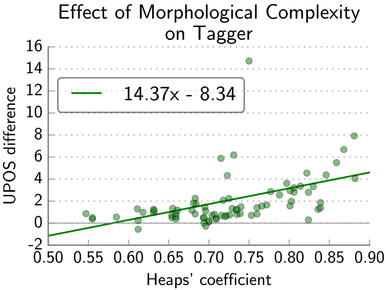
- When using in the parser/tagger, add with pretrained and frequent-token embeddings (eltwise)



Character model: Experiment

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





Results

Results

Treebanks	**UPOS**	**XPOS**	**UAS**	**LAS**	**CLAS**
All treebanks	**93.09**	**82.27**	**81.30**	**76.30**	**72.57**
Large treebanks	**95.58**	**94.56**	**85.16**	**81.77**	**78.40**
Parallell treebanks	**88.25**	30.66	**80.17**	**73.73**	**69.88**
Small treebanks	**87.02**	**82.03**	70.19	61.02	54.76
Surprise treebanks	–	–	54.47	40.57	37.41

System	**UPOS**	**XPOS**	**UAS**	**LAS**	**CLAS**
Dozat et al.	**93.09**	**82.27**	**81.30**	**76.30**	**72.57**
Björkelund et al.	*91.98*	64.84	79.90	74.42	70.18
Yu et al.	91.00	*79.93*	74.22	68.41	63.24
Shi et al.	90.88	79.80	*80.35*	*75.00*	*70.91*

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 that were used in the minibatch
  - I.e. for words that *are* attested in the minibatch, we apply Adam's accumulator update rule:

$$\mathbf{m}_t = \beta_1 \mathbf{m}_{t-1} + (1 - \beta_1)\mathbf{g}_t$$
$$\mathbf{v}_t = \beta_2 \mathbf{v}_{t-1} + (1 - \beta_2)\mathbf{g}_t^2$$

  - 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

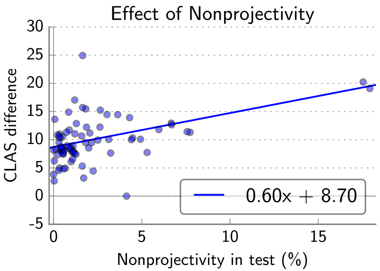
- Preference for initializing to zero wherever possible
  - Bias terms
  - Final linear layers (character model, output layers)
  - Word/POS embeddings (other than pretrained)
- Otherwise, we use orthonormal initialization (Saxe et al., 2014)

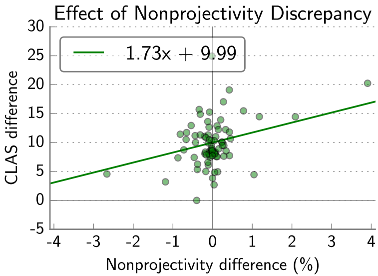
Recurrent Cells

- 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 I

Ballesteros, M., Dyer, C., and Smith, N. A. (2015). Improved transition-based parsing by modeling characters instead of words with lstms. *EMNLP*.

Björkelund, A., Falenska, A., Yu, X., and Kuhn, J. (2017). Ims at the conll 2017 ud shared task: Crfs and perceptrons meet neural networks. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 40–51, Vancouver, Canada. Association for Computational Linguistics.

Cao, K. and Rei, M. (2016). A joint model for word embedding and word morphology. *ACL 2016*, page 18.

References II

Dozat, T. and Manning, C. D. (2017). Deep biaffine attention for neural dependency parsing. *ICLR 2017*.

Dozat, T., Qi, P., and Manning, C. D. (2017). Stanford's graph-based neural dependency parser at the conll 2017 shared task. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 20–30, Vancouver, Canada. Association for Computational Linguistics.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. *International Conference on Machine Learning*.

References III

Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*.

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. *International Conference on Learning Representations*.

Krueger, D., Maharaj, T., Kramár, J., Pezeshki, M., Ballas, N., Ke, N. R., Goyal, A., Bengio, Y., Larochelle, H., Courville, A., et al. (2017). Zoneout: Regularizing rnns by randomly preserving hidden activations. *ICLR 2017*.

Saxe, A. M., McClelland, J. L., and Ganguli, S. (2014). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *ICLR 2014*, abs/1312.6120.

References IV

Shi, T., Wu, F. G., Chen, X., and Cheng, Y. (2017). Combining global models for parsing universal dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 31–39, Vancouver, Canada. Association for Computational Linguistics.

Straka, M., Hajic, J., Straková, J., and Hajic jr, J. (2015). Parsing universal dependency treebanks using neural networks and search-based oracle. In *International Workshop on Treebanks and Linguistic Theories (TLT14)*, page 208.

References V

Yu, K., Sofroniev, P., Schill, E., and Hinrichs, E. (2017). The parse is darc and full of errors: Universal dependency parsing with transition-based and graph-based algorithms. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 126–133, Vancouver, Canada. Association for Computational Linguistics.