Deep Biaffine Attention for Neural Dependency Parsing

Timothy Dozat Christopher D. Manning

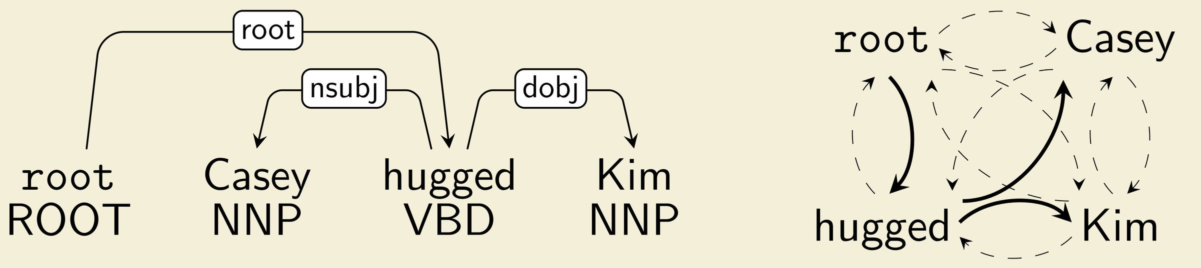
Stanford University

Goals

- Much research has been devoted to developing neural dependency parsers with complex, task-specific architecture
- Typical approach: use specialized neural networks to predict discrete actions in a dedicated, transition-based parsing algorithm
  SyntaxNet AKA Parsey McParseface (Andor et al., 2016): Feedforward network with beam search and CRF loss
  Ablated RNN Grammar (Kuncoro et al., 2016): Stack-LSTM with bidirectional LSTM for phrase composition (SOTA)
- **Can we get competitive (or even superior) parsing results with a simple architecture using general-purpose components?**

Dependency Parsing

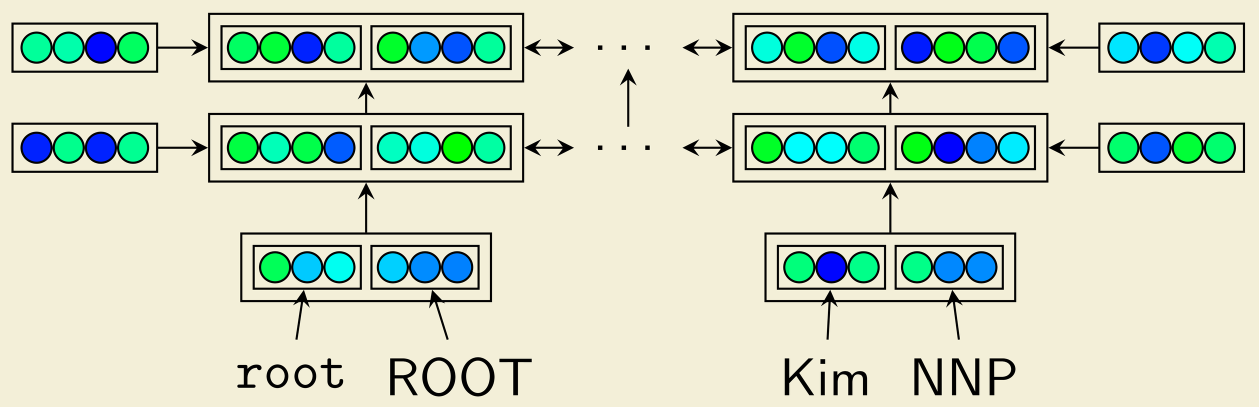
- Automatically annotate sentences, focusing on the functional role each phrase plays
  Head: Edge source, more contentful role (predicate → arguments)
  Dependent: Edge target
  Label: Edge type (*N*ominal *SUBJ*ect, *A*djectival *CL*ause)



- Particularly useful for NLU tasks, such as semantic parsing or knowledge base population
- Graph-based approach to parsing: assign weights to each possible edge, construct a maximum spanning tree

LSTM

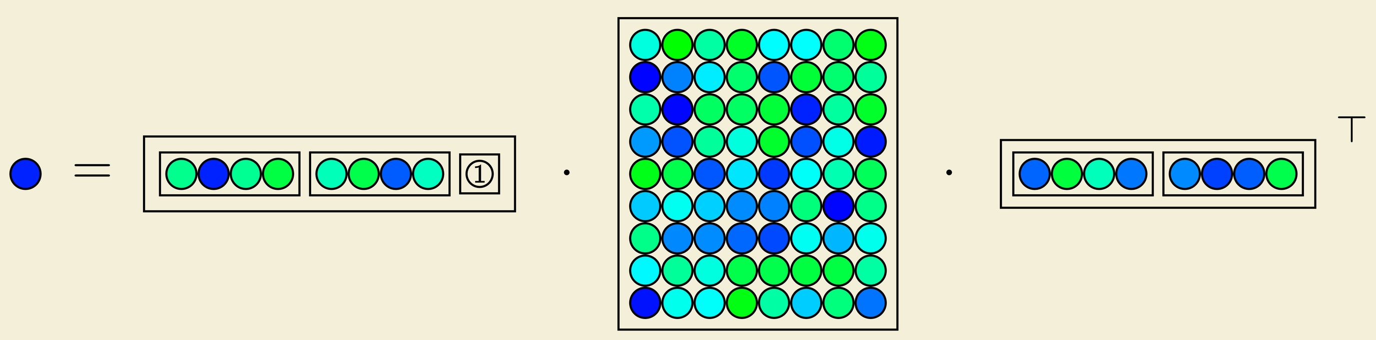
Step one: BiLSTM over the sequence of word and part of speech tag embeddings, take all topmost LSTM states R ($=\text{stack}_{i=1}^{n}(\mathbf{r}_i)$)



Variable-class classification (= attention)

- We want to predict heads (classes) given dependents (inputs), but the number of possible heads changes from sentence to sentence
- Thus, we want to predict $P(y_i^{(edge)} = j|\mathbf{r}_i; \mathbf{r}_j)$
- softmax$(RU^{(1)}\mathbf{r}_i + R\mathbf{u}^{(2)})$ achieves this naturally

$$P(j|\mathbf{r}_i; \mathbf{r}_j) \propto \exp\left(\mathbf{r}_i^\top U^{(1)}\mathbf{r}_j\right) \exp\left(\mathbf{u}^{\top(2)}\mathbf{r}_j\right)$$



- After deciding on an edge from j to i, we want to predict the label
- This time, we want to predict $P(y^{(label)} = l|\mathbf{r}_i, \mathbf{r}_{y_i^{(edge)}})$
- We can use softmax$(\mathbf{r}_{y_i}^\top \mathbf{U}^{(1)}\mathbf{r}_i + U^{(2)}(\mathbf{r}_{y_i} \oplus \mathbf{r}_i) + \mathbf{b})$ to model this

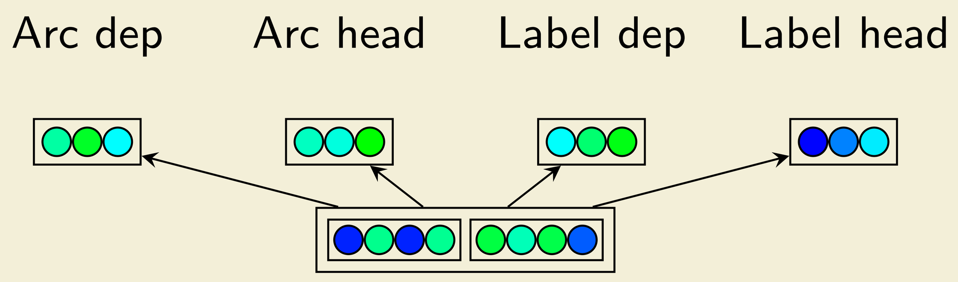
$$P(l|\mathbf{r}_i, \mathbf{r}_{y_i^{(edge)}}) \propto \exp\left(\mathbf{r}_i^\top U_l^{(1)}\mathbf{r}_{y_i}\right) \exp\left(\mathbf{r}_i^\top \mathbf{u}_l^{(2)}\right) \exp\left(\mathbf{r}_{y_i}^\top \mathbf{u}_l^{(3)}\right) \exp\left(b_l\right)$$

- Closely related to linear models with interactions

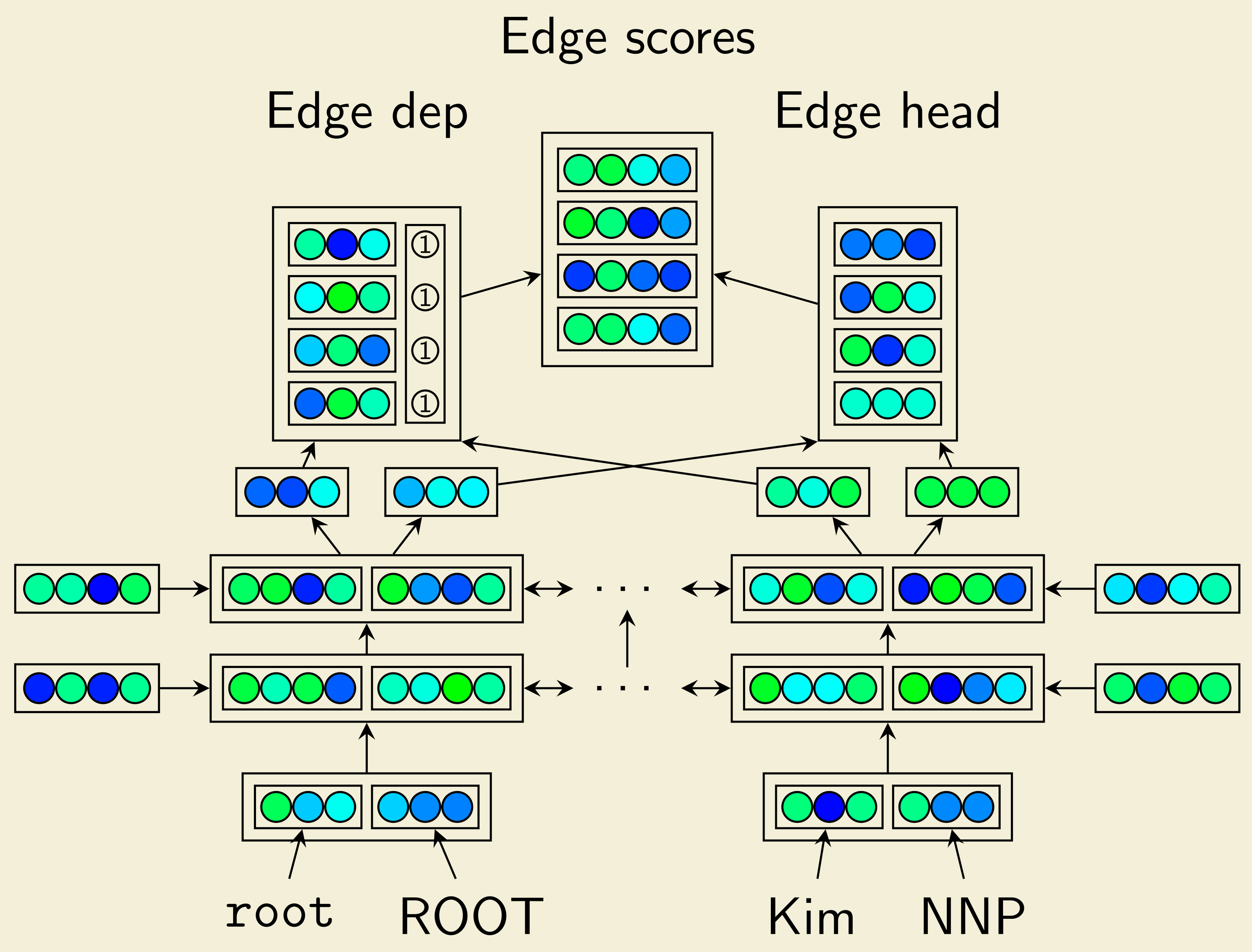
```
scores ~ head.vector * dep.vector
```

Practical modifications

- Everything is so big!
- We can get more control over the tradeoffs between speed, overfitting, and underfitting by shrinking $\mathbf{r}_i$ with smaller MLPs before the biaffine output layers (*deep biaffine* model as opposed to *shallow biaffine*)
- Result: four representations for each word
- Naturally reflects the intuition that the relationships we want to capture are asymmetric



Final model (edge scorer)



Hyperparameters

Param	Value	Param	Value
Embedding size	100	Embedding dropout	33%
LSTM size	400	LSTM dropout	33%
Edge MLP size	500	Edge MLP dropout	33%
Label MLP size	100	Label MLP dropout	33%
LSTM depth	3	MLP depth	1
α	$2e^{-3}$	β_1,β_2	.9
Annealing	$.75^{\frac{t}{5000}}$	t_{max}	50,000

- Relatively large network (other models use ∼ 100 LSTM dims)
- Highly regularized with dropout
- Reducing *Adam*'s β_2 from .999 to .9 significantly improved performance ($p < .05$)

Related work

- Transition-based
  Nivre et al. (2006): Feature-based
  Chen and Manning (2014): First successful neural parser
  Andor et al. (2016): Extend with beam search / CRF loss
  Kuncoro et al. (2016): Extend with LSTMs (SOTA)
- Graph-based
  McDonald and Pereira (2006): Feature-based
  Kiperwasser and Goldberg (2016): First neural graph-based parser
  Cheng et al. (2016): Keep track of previous decisions
  Hashimoto et al. (2016): Jointly learn tagging & chunking

PTB Results

Type	Model	SD 3.3.0 UAS	SD 3.3.0 LAS	CTB UAS	CTB LAS
Transition	Ballesteros et al. (2016)	93.6	91.4	87.7	86.2
	Andor et al. (2016)	94.6	92.8	–	–
	Kuncoro et al. (2016)	**95.8**	**94.6**	–	–
Graph	Kiperwasser and Goldberg (2016)	93.9	91.9	87.6	86.1
	Cheng et al. (2016)	94.1	91.5	88.1	85.7
	Hashimoto et al. (2016)	94.7	92.9	–	–
	Deep biaffine	95.7	94.1	**89.3**	**88.2**

CoNLL 09 Results

Model	Catalan UAS	Catalan LAS	Chinese UAS	Chinese LAS	Czech UAS	Czech LAS
Andor et al.	92.7	89.8	84.7	80.9	88.9	84.6
Deep biaffine	**94.7**	**92.0**	**88.9**	**85.4**	**92.1**	**87.4**

Model	English UAS	English LAS	German UAS	German LAS	Spanish UAS	Spanish LAS
Andor et al.	93.2	91.2	90.9	89.2	92.6	90.0
Deep biaffine	**95.2**	**93.2**	**93.5**	**91.4**	**94.3**	**91.7**

Affect of classifier type (SD 3.5.0)

Model	Classifier UAS	Classifier LAS	Classifier Sents/sec
Deep biaffine	**95.8**	**94.2**	**410.9**
Shallow biaffine	95.7	94.0*	299.0
Shallow b. (50% MLP dropout)	95.7	94.1*	300.1
Shallow b. (300d LSTM)	95.6*	93.9*	373.2
Traditional attention	95.5*	93.9*	367.4

(Statistical significances are marked with an asterisk)

Conclusion

- Our simple, straightforward parser uses only neural components, effectively no task-specific architecture
- **Substantially outperforms most more complex neural transition-based parsers**
- **Substantially outperforms all other neural graph-based parsers**
- The biaffine approach to attention is theoretically justified, here beats the more traditional approach
- Adding final MLP layers to the LSTM helps to maximize speed and performance, captures head-dependent asymmetries
- This work provides a fast, simple, high-performing baseline against which to test more complex architectures

References

Andor, D., Alberti, C., Weiss, D., Severyn, A., Presta, A., Ganchev, K., Petrov, S., and Collins, M. (2016). Globally normalized transition-based neural networks. In *Association for Computational Linguistics*.

Ballesteros, M., Goldberg, Y., Dyer, C., and Smith, N. A. (2016). Training with exploration improves a greedy stack-LSTM parser. *Proceedings of the conference on empirical methods in natural language processing*.

Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 740–750.

Cheng, H., Fang, H., He, X., Gao, J., and Deng, L. (2016). Bi-directional attention with agreement for dependency parsing. *arXiv preprint arXiv:1608.02076*.

Hashimoto, K., Xiong, C., Tsuruoka, Y., and Socher, R. (2016). A joint many-task model: Growing a neural network for multiple nlp tasks. *arXiv preprint arXiv:1611.01587*.

Kiperwasser, E. and Goldberg, Y. (2016). Simple and accurate dependency parsing using bidirectional LSTM feature representations. *Transactions of the Association for Computational Linguistics*, 4:313–327.

Kuncoro, A., Ballesteros, M., Kong, L., Dyer, C., Neubig, G., and Smith, N. A. (2016). What do recurrent neural network grammars learn about syntax? *CoRR*, abs/1611.05774.

McDonald, R. T. and Pereira, F. C. (2006). Online learning of approximate dependency parsing algorithms. In *EACL*.

Nivre, J., Hall, J., and Nilsson, J. (2006). Maltparser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC*, volume 6, pages 2216–2219.

tdozat@stanford.edu