Neural CRF Parsing

AUTHORS: GREG DURRETT AND DAN KLEIN

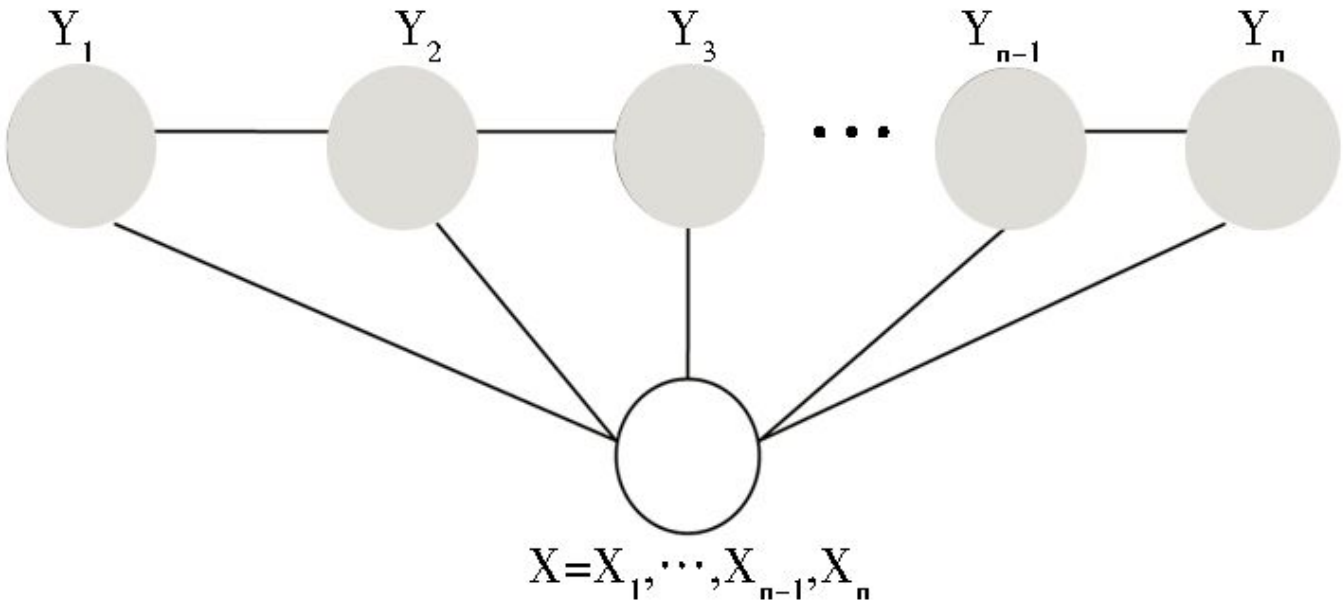
PRESENTER: YUNDI FEI

Overview

- Based on the baseline CRF model by Hall et al. (2014)
- What is CRF(Conditional Random Field)?
  - A class of statistical modeling method in the sequence modeling family
  - Often used for labeling or parsing of sequential data, such as natural language processing

Overview

- What is CRF(Conditional Random Field)?
  - Defines posterior probability of a label sequence given an input observation sequence
    - Conditional probability is P(label sequence Y | observation sequence X) instead of P(Y | X)
    - The probability of a transition between the labels may depend on past and future observations



Xu, D. (2017, March). [CRF Introduction Slide]. Retrieved March 6, 2018, from http://images.slideplayer.com/35/10389057/slides/slide_3.jpg

Overview

- This work: a CRF constituency parser that individual anchored rule productions are scored based on **nonlinear features computed with a feedforward neural network** in addition to **linear functions of sparse indicator** features like a standard CRF

Prior work

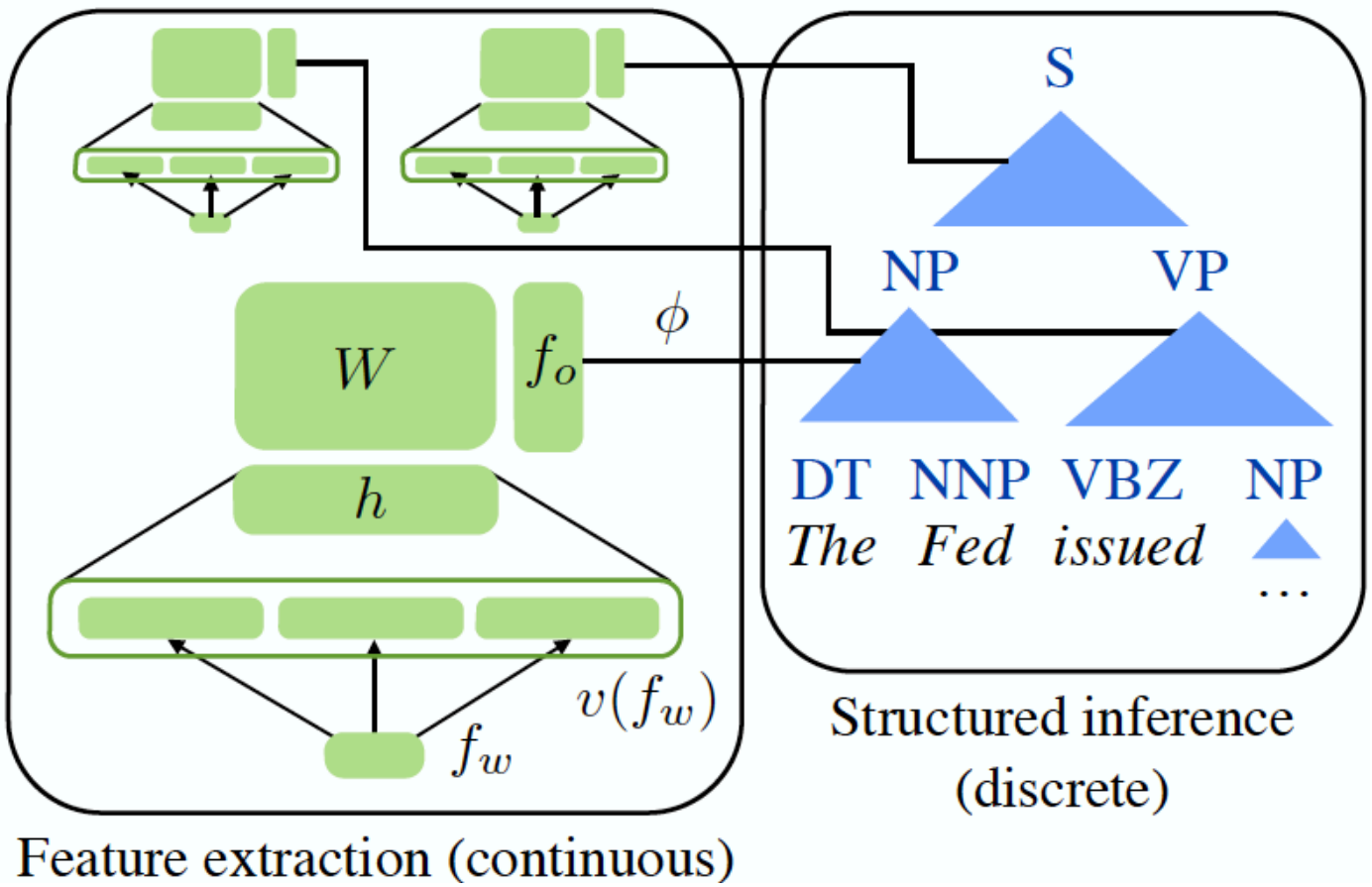
- Compared to conventional CRFs
  - Scores can be thought of as **nonlinear potentials analogous** to linear potentials in conventional CRFs
  - Computations factor along the same substructures as in standard CRFs

Prior work

- Compared to prior neural network models
  - **Prior: sidestepped the problem of structured inference** by making sequential decisions or by doing reranking
  - **This framework: permits exact inference via CKY**, since the model's structured interactions are purely discrete and do not involve continuous hidden state

Model

- Decomposes over anchored rules, and it scores each to these with a potential function
  - **Nonlinear functions of word embedding**
  - **Linear functions of sparse indicator** features like a standard CRF



Model – Anchored Rule

- **An anchored rule: a tuple** (r, s)
  - r = an indicator of the rule's identity
  - s = (i, j, k) indicator of the span (i, k) and split point j of the rule
    - For unary rules, specify a null value for the split point

Model – Anchored Rule

- A tree T = a collection of anchored rules subject to the constraint that those rules form a tree
- All of the parsing models are CRFs that decompose over anchored rule productions and place a probability distribution over trees conditioned on a sentence w as

$$P(T|\mathbf{w}) \propto \exp\left(\sum_{(r,s)\in T} \phi(\mathbf{w}, r, s)\right)$$

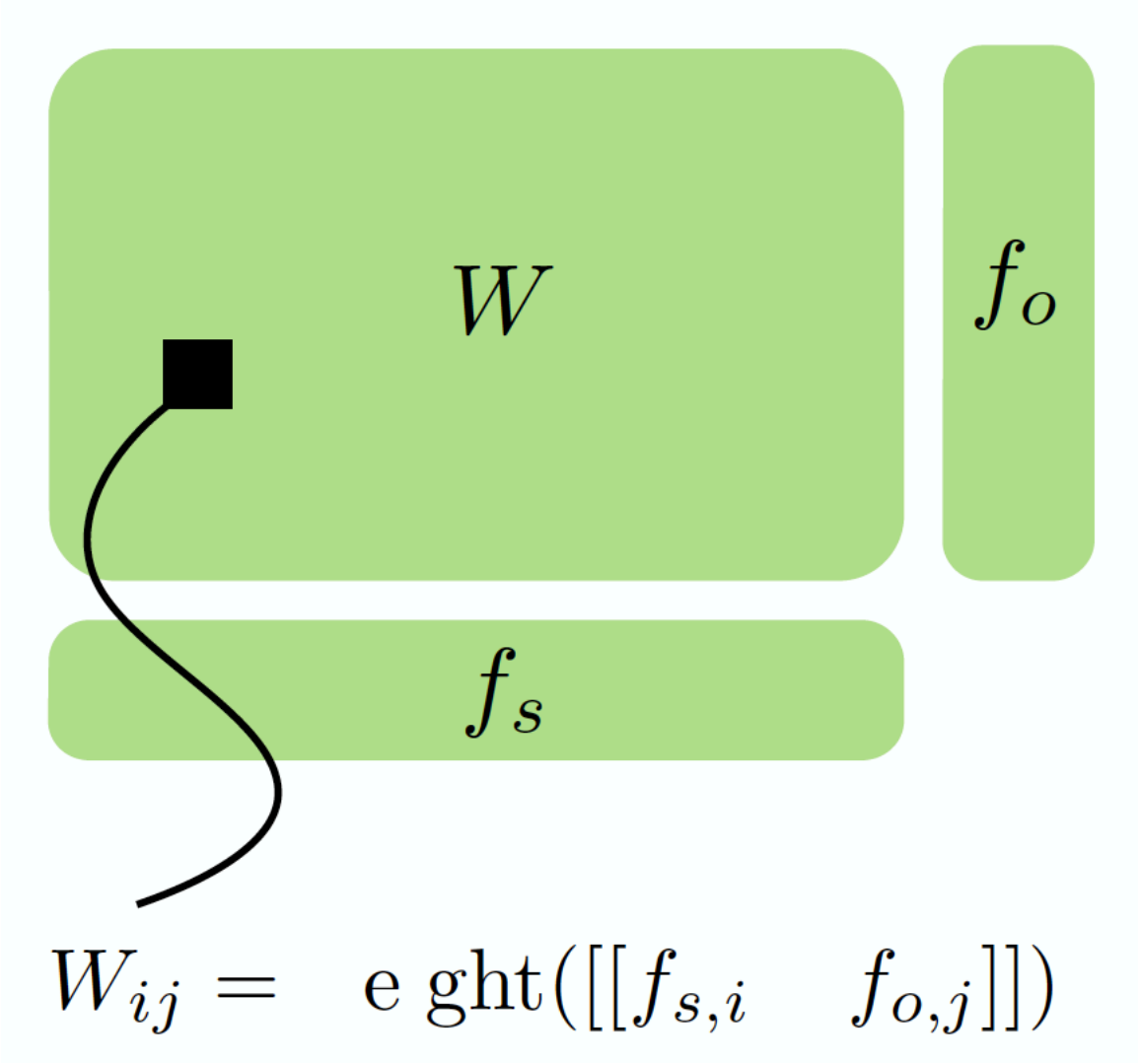
Model – Scoring Anchored Rule

- Φ is a scoring function that considers the input sentence and the anchored rule
  - Can be a neural net, a linear function of surface features, or combination of the two

Model – Scoring Anchored Rule

- Baseline sparse scoring function
  - $f_o(r) \in \{0,1\}^{n_o}$ a sparse vector of features expressing properties of r (such as the rule's identity or its parent label)
  - $f_s(\boldsymbol{w}, s) \in \{0,1\}^{n_s}$ a sparse vector of surface features associated with the words in the sentence and the anchoring
  - W a $n_s \times n_o$ matrix of weights

$$\phi_{\text{sparse}}(\mathbf{w}, r, s; W) = f_s(\mathbf{w}, s)^\top W f_o(r)$$

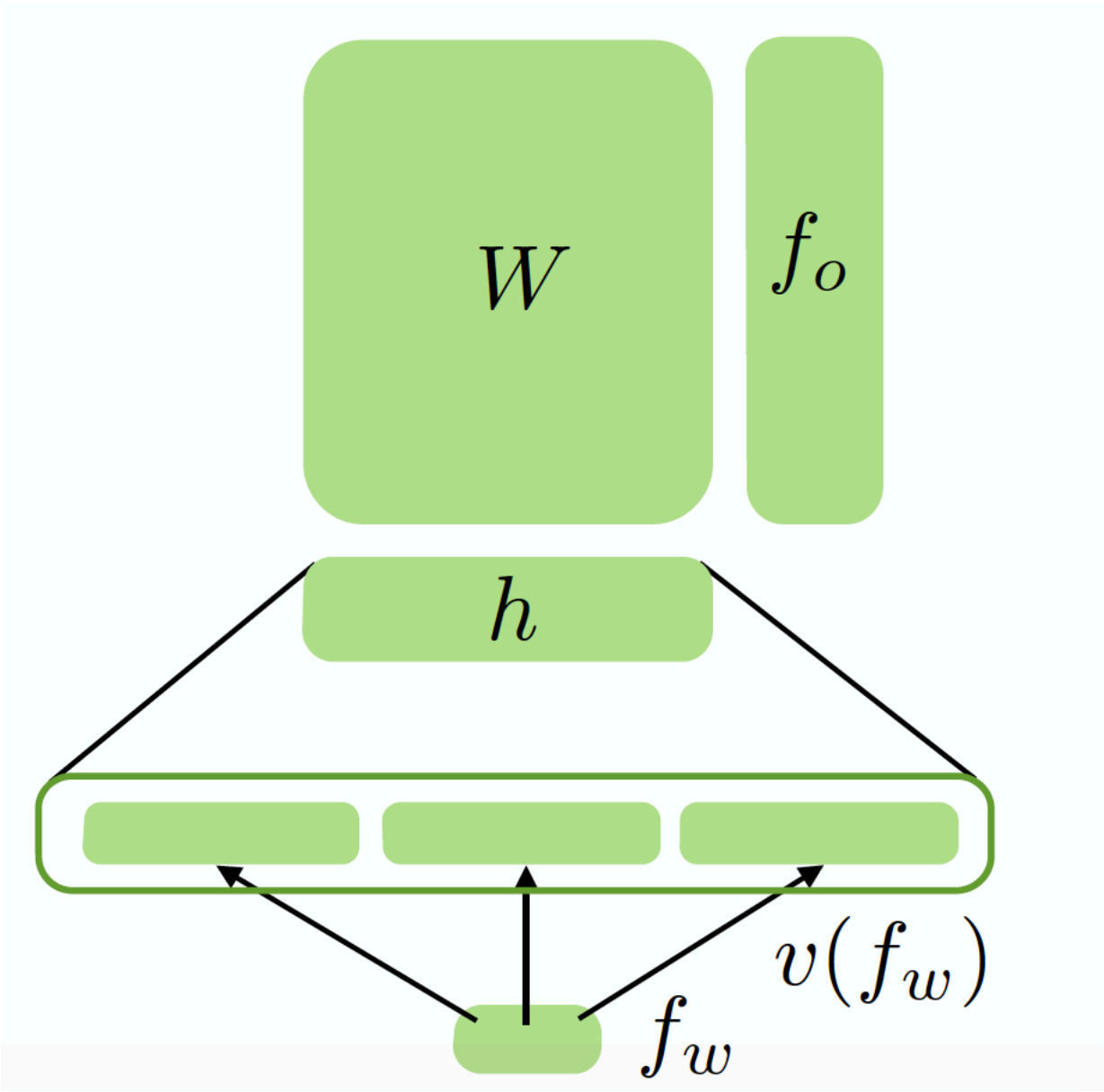

W
f_o
f_s
$W_{ij} = \quad \text{e ght}([[f_{s,i} \quad f_{o,j}]])$

Model – Scoring Anchored Rule

- Neural scoring function
  - $f_w(\boldsymbol{w}, s) \in \mathbb{N}^{n_w}$ a function that produces a fixed-length sequence of word indicators based on the input sentence and the anchoring
  - $v : \mathbb{N} \rightarrow \mathbb{R}^{n_e}$ embedding function
    - the dense representations of the words are subsequently concatenated to form a vector we denote by $v(f_w)$
  - $H \in \mathbb{R}^{n_h \times (n_w n_e)}$ real valued parameters
  - g elementwise nonlinearity: rectified linear units $g(x) = \max(x, 0)$

$$h(\mathbf{w}, s; H) = g(Hv(f_w(\mathbf{w}, s)))$$

$$\phi_{\text{neural}}(\mathbf{w}, r, s; H, W) = h(\mathbf{w}, s; H)^\top W f_o(r)$$


W
f_o
h
$v(f_w)$
f_w

Model – Scoring Anchored Rule

- Two models combined:

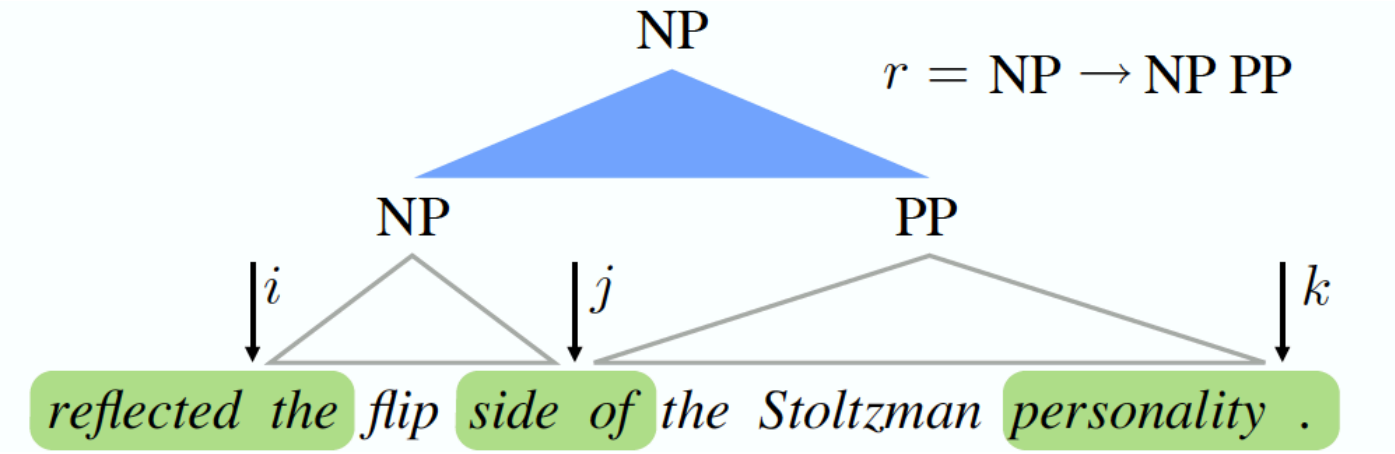
$$\phi(\mathbf{w}, r, s; W_1, H, W_2) = \phi_{\text{sparse}}(\mathbf{w}, r, s; W_1) + \phi_{\text{neural}}(\mathbf{w}, r, s; H, W_2)$$

Features

- Sparse: f_s
  - At **pretermimal** layer
    - Prefixes and suffixes up to length 5 of the current word and neighboring words as well as the words' identities
  - For **nonterminal** productions, fire indicators on
    - The words before and after the start, end, and split point of the anchored rule
    - Span properties: span length + span shape
      - Span shape: an indicator of where capitalized words, numbers, and punctuation occur in the span

Features

- Neural: f_w
  - Words surrounding the beginning and end of a span and the split point
    - Look two words in either direction around each point of interest



- Neural: v
  - Use pre-trained word vectors from Bansal et al. (2014)
    - Contrary to standard practice, do not update these vectors during training

Learning

- To learn weights for neural model, maximize the conditional log likelihood of D training trees T^*

$$\mathcal{L}(H, W) = \sum_{i=1}^{D} \log P(T_i^* | \mathbf{w}_i; H, W)$$

Learning

- **The gradient of W takes** the standard form of log-linear models:

$$\frac{\partial \mathcal{L}}{\partial W} = \left(\sum_{(r,s) \in T^*} h(\mathbf{w}, s; H) f_o(r)^\top \right) - \left(\sum_T P(T|\mathbf{w}; H, W) \sum_{(r,s) \in T} h(\mathbf{w}, s; H) f_o(r)^\top \right)$$

Learning

- To **update $\boldsymbol{H}$**, use standard backpropagation
  - First, compute

$$\frac{\partial \mathcal{L}}{\partial h} = \left(\sum_{(r,s) \in T^*} W f_o(r) \right) - \left(\sum_{T} P(T|\mathbf{w}; H, W) \sum_{(r,s) \in T} W f_o(r) \right)$$

  - Because h is the output of the neural network, then apply the chain rule to compute gradients for H and any other parameters in the neural network

Learning

- **Momentum term $\rho = \mathbf{0.95}$** as suggested by Zeiler (2012)
- **Use minibatch size of 200 trees**
  - For each treebank, train for either 10 passes through the treebank or 1000 minibatches, whichever is shorter
- **Initialize the output weight matrix W to zero**
- **Initialize the lower level neural network parameters H** with each entry being **independently sampled from a Gaussian** with mean 0 and variance 0.01
  - Better than uniform initialization but variance not important

Improvements

- Follow Hall et al. (2014) and prune according to an X-bar grammar with head-outward binarization
  - Ruling out any constituent whose max marginal probability is less than e^{-9}
  - Reduce the number of spans and split points

Improvements

- Note that the same word will appear in the same position in a large number of span/split point combinations, and cache the contributions to the hidden layer caused by that word (Chen and Manning, 2014)

Test results

- Section 23 of the **English** Penn Treebank (PTB)

	F_1 all
Single model, PTB only	
Hall et al. (2014)	89.2
Berkeley	90.1
Carreras et al. (2008)	91.1
Shindo et al. (2012) single	91.1
Single model, PTB + vectors/clusters	
Zhu et al. (2013)	91.3
This work*	91.1

	F_1 all
Extended conditions	
Charniak and Johnson (2005)	91.5
Socher et al. (2013)	90.4
Vinyals et al. (2014) single	90.5
Vinyals et al. (2014) ensemble	91.6
Shindo et al. (2012) ensemble	92.4

Test results

- on the **nine languages** used in the SPMRL 2013 and 2014 shared tasks

	Arabic	Basque	French	German	Hebrew	Hungarian	Korean	Polish	Swedish	Avg
					Dev, all lengths					
Hall et al. (2014)	78.89	83.74	79.40	83.28	88.06	87.44	81.85	91.10	75.95	83.30
This work*	**80.68**	**84.37**	**80.65**	**85.25**	**89.37**	**89.46**	**82.35**	**92.10**	**77.93**	**84.68**
					Test, all lengths					
Berkeley	79.19	70.50	80.38	78.30	86.96	81.62	71.42	79.23	79.18	78.53
Berkeley-Tags	78.66	74.74	79.76	78.28	85.42	85.22	78.56	86.75	80.64	80.89
Crabbé and Seddah (2014)	77.66	85.35	79.68	77.15	86.19	87.51	79.35	91.60	82.72	83.02
Hall et al. (2014)	78.75	83.39	79.70	78.43	87.18	88.25	80.18	90.66	82.00	83.17
This work*	**80.24**	**85.41**	**81.25**	**80.95**	**88.61**	**90.66**	**82.23**	**92.97**	**83.45**	**85.08**
					Reranked ensemble					
2014 Best	**81.32**	**88.24**	**82.53**	**81.66**	**89.80**	**91.72**	**83.81**	**90.50**	**85.50**	**86.12**

Conclusion

- Decomposes over anchored rules, and it scores each with a potential function
- Add scoring based on nonlinear features computed with a feedforward neural network to the baseline CRF model
- Improvements for both English and 9 other languages

Thank you!