Structured Learning with Inexact Search: Advances in Shift-Reduce CCG Parsing

This work is made possible and fully supported by the Carnegie Trust for the Universities of Scotland and the Cambridge Trust.
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in \mathcal{Y}_x} \sum_{d \in \mathcal{D}(y)} \text{score}(\Phi(d)) \]
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in \mathcal{Y}_x} \sum_{d \in \mathcal{D}(y)} \text{score}(\Phi(d)) \]

- Decomposition: \( \mathcal{D}(y) \)
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in \mathcal{Y}_x} \sum_{d \in \mathcal{D}(y)} \text{score}(\Phi(d)) \]

- Decomposition: \( \mathcal{D}(y) \)
- Scoring: \( \text{score}(\Phi(d)) \)
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in \mathcal{Y}_x} \sum_{d \in \mathcal{D}(y)} \text{score}(\Phi(d)) \]

- Decomposition: \( \mathcal{D}(y) \)
- Scoring: \( \text{score}(\Phi(d)) \)
- Summing: \( \sum_{d \in \mathcal{D}(y)} \)
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in \mathcal{Y}_x} \sum_{d \in \mathcal{D}(y)} \text{score}(\Phi(d)) \]

- **Decomposition**: \( \mathcal{D}(y) \)
- **Scoring**: \( \text{score}(\Phi(d)) \)
- **Summing**: \( \sum_{d \in \mathcal{D}(y)} \)
- **Search**: \( \arg \max_{y \in \mathcal{Y}_x} \)
Structured Prediction in NLP

\[ y^* = \arg \max_{y \in Y_x} \sum_{d \in D(y)} \text{score}(\Phi(d)) \]

- Decomposition: \( D(y) \)
- Scoring: \( \text{score}(\Phi(d)) \)
- Summing: \( \sum_{d \in D(y)} \)
- Search: \( \arg \max_{y \in Y_x} \)

\( Y_x \) is exponentially-sized and prohibitive to enumerate.
Structured Prediction: Sequence Labelling
MEMM [McCallum et al., 2000]

\[ p(y_1, \ldots, y_m | x_1, \ldots, x_m) \]
MEMM \[\text{McCallum et al., 2000}\]

\[
p(y_1, \ldots, y_m | x_1, \ldots, x_m) = \prod_{i=1}^{m} p(y_i | y_1, \ldots, y_{i-1}, x_1, \ldots, x_m)
\]
MEMM [McCallum et al., 2000]

\[ p(y_1, \ldots, y_m | x_1, \ldots, x_m) \]

\[ = \prod_{i=1}^{m} p(y_i | y_1 \ldots, y_{i-1}, x_1, \ldots, x_m) \]

\[ = \prod_{i=1}^{m} p(y_i | y_{i-1}, x_1, \ldots, x_m) \]
MEMM [McCallum et al., 2000]

\[ p(y_1, \ldots, y_m | x_1, \ldots, x_m) \]

\[ = \prod_{i=1}^{m} p(y_i | y_1, \ldots, y_{i-1}, x_1, \ldots, x_m) \]

\[ = \prod_{i=1}^{m} p(y_i | y_{i-1}, x_1, \ldots, x_m) \]

\[ = \prod_{i=1}^{m} \frac{\exp\{\mathbf{w} \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i)\}}{\sum_{y_i'} \exp\{\mathbf{w} \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i')\}} \]
CRF [Lafferty et al., 2001]

\[ p(y_1, \ldots, y_m | x_1, \ldots, x_m) \]
CRF \[\text{[Lafferty et al., 2001]}\]

\[
p(y_1, \ldots, y_m | x_1, \ldots, x_m) = \frac{1}{Z} \exp \left\{ \sum_{i=1}^{m} \sum_{j=1}^{F} \omega_j \phi_j(y_{i-1}, y_i, x, i) \right\}
\]
CRF [Lafferty et al., 2001]

\[ p(y_1, \ldots, y_m \mid x_1, \ldots, x_m) \]

\[ = \frac{1}{z} \exp \left\{ \sum_{i=1}^{m} \sum_{j=1}^{F} \omega_j \phi_j (y_{i-1}, y_i, x, i) \right\} \]

\[ z = \sum_{y_1:m \in \mathcal{Y}_x} \exp \left\{ \sum_{i=1}^{m} \sum_{j=1}^{F} \omega_j \phi_j (y_{i-1}, y_i, x, i) \right\} \]
MEMM and CRF

\[ p(y_1, \ldots, y_m|x_1, \ldots, x_m) = \prod_{i=1}^{m} \frac{\exp\{w \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i)\}}{\sum_{y_i'} \exp\{w \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i')\}} \]

\[ p(y_1, \ldots, y_m|x_1, \ldots, x_m) = \frac{1}{z} \exp\{\sum_{i=1}^{m} \sum_{j=1}^{F} \omega_j \phi_j(y_{i-1}, y_i, x, i)\} \]

- Feature function: \( \Phi \)
- Structured output: \( Y \)
- Search: dynamic programming + Viterbi decoding
- \( \arg \max_{y_{1:m}} p(y_1, \ldots, y_m|x_1, \ldots, x_m) \)
MEMM and CRF

\[ p(y_1, \ldots, y_m|x_1, \ldots, x_m) = \prod_{i=1}^{m} \frac{\exp \{ w \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i) \}}{\sum_{y_i'} \exp \{ w \cdot \Phi(x_1, \ldots, x_m, i, y_{i-1}, y_i') \}} \]

\[ p(y_1, \ldots, y_m|x_1, \ldots, x_m) = \frac{1}{Z} \exp \left\{ \sum_{i=1}^{m} \sum_{j=1}^{F} \omega_j \phi_j(y_{i-1}, y_i, x, i) \right\} \]

- Feature function: \( \Phi \)
- Structured output: \( y \)
- Search: dynamic programming + Viterbi decoding
- \( \arg \max_{y_{1:m}} p(y_1, \ldots, y_m|x_1, \ldots, x_m) \)
The Structured Perceptron [Collins, 2002]

1: \( w \leftarrow 0 \) \hspace{1cm} \triangleright \text{the input is the training set } \{(x_i, y_i)\}_{i=1}^n
2: \textbf{while} \text{ not converged do}
3: \textbf{for} \ i \leftarrow 1, \ldots, n \ \textbf{do}
4: \quad y^* \leftarrow \arg \max_{y \in \text{GEN}(x_i)} w \cdot \Phi(x_i, y) \hspace{1cm} \triangleright \text{obtain model prediction}
5: \quad \textbf{if} \ y^* \neq y_i \ \textbf{then} \hspace{1cm} \triangleright \text{y}^* \text{ not correct}
6: \quad w \leftarrow w + \Phi(x_i, y_i) - \Phi(x_i, y^*) \hspace{1cm} \triangleright \text{online update}
### The Structured Perceptron [Collins, 2002]

1. \( \mathbf{w} \leftarrow 0 \) \hspace{1cm} \triangleright \text{the input is the training set } \{(x_i, y_i)\}_{i=1}^n

2. \textbf{while} not converged \textbf{do}

3. \textbf{for} \( i \leftarrow 1, \ldots, n \) \textbf{do}

4. \( y^* \leftarrow \arg \max_{y \in \text{GEN}(x_i)} \mathbf{w} \cdot \Phi(x_i, y) \) \hspace{1cm} \triangleright \text{obtain model prediction}

5. \textbf{if} \( y^* \neq y_i \) \textbf{then}

6. \( \mathbf{w} \leftarrow \mathbf{w} + \Phi(x_i, y_i) - \Phi(x_i, y^*) \) \hspace{1cm} \triangleright \text{y* not correct}

\hspace{1cm} \triangleright \text{online update}
The Structured Perceptron [Collins, 2002]

1: \( \mathbf{w} \leftarrow 0 \)  \hspace{1cm} \triangleright \text{the input is the training set } \{(x_i, y_i)\}_{i=1}^n \\
2: \textbf{while} \text{ not converged do} \\
3: \textbf{for} i \leftarrow 1, \ldots, n \textbf{ do} \\
4: \quad y^* \leftarrow \arg \max_{y \in \text{GEN}(x_i)} \mathbf{w} \cdot \Phi(x_i, y) \quad \hspace{1cm} \triangleright \text{obtain model prediction} \\
5: \quad \textbf{if} \ y^* \neq y_i \textbf{ then} \\
6: \quad \mathbf{w} \leftarrow \mathbf{w} + \Phi(x_i, y_i) - \Phi(x_i, y^*) \hspace{1cm} \triangleright \text{online update} \\

- Feature function: \( \Phi \) \\
- Structured output: \( Y \) \\
- Search: dynamic programming
The Structured Perceptron [Collins, 2002]

1: \( \mathbf{w} \leftarrow 0 \) \hspace{1cm} \triangleright \text{the input is the training set } \{(x_i, y_i)\}_{i=1}^{n}
2: \textbf{while} \text{ not converged \textbf{do}}
3: \textbf{for} \ i \leftarrow 1, \ldots, n \ \textbf{do}
4: \quad y^* \leftarrow \arg \max_{y \in \text{GEN}(x_i)} \mathbf{w} \cdot \Phi(x_i, y) \quad \triangleright \text{obtain model prediction}
5: \quad \textbf{if} \ y^* \neq y_i \ \textbf{then}
6: \quad \quad \mathbf{w} \leftarrow \mathbf{w} + \Phi(x_i, y_i) - \Phi(x_i, y^*) \quad \triangleright \text{online update}

- Feature function: \( \Phi \)
- Structured output: \( Y \)
- Search: dynamic programming
  - beam search (the incremental structured perceptron [Collins and Roark, 2004])
The Structured Perceptron [Collins, 2002]

1. \( w \leftarrow 0 \) \quad \triangleright \text{the input is the training set } \{(x_i, y_i)\}_{i=1}^{n}
2. \text{while not converged do}
3. \text{for } i \leftarrow 1, \ldots, n \text{ do}
4. \quad y^* \leftarrow \arg \max_{y \in \text{GEN}(x_i)} w \cdot \Phi(x_i, y) \quad \triangleright \text{obtain model prediction}
5. \quad \text{if } y^* \neq y_i \text{ then} \quad \triangleright y^* \text{ not correct}
6. \quad w \leftarrow w + \Phi(x_i, y_i) - \Phi(x_i, y^*) \quad \triangleright \text{online update}

- Feature function: \( \Phi \)
- Structured output: \( Y \)
- Search: dynamic programming
  - beam search (the incremental structured perceptron [Collins and Roark, 2004])
  - dynamic programming + cube pruning [Chiang, 2007]
Structured Perceptron with Inexact Search [Huang et al., 2012]

Graph-based dependency parsing

[Zhang and McDonald, 2012; Zhang et al., 2013]
Structured Perceptron with Inexact Search [Huang et al., 2012]

Hierarchical phrase-based translation [Zhao et al., 2014]
Neural Network Models

• Sequence-to-Sequence [Sutskever et al., 2014]
  
  – training: per-step cross-entropy
  
  – test: \( p(y_1, \ldots, y_n|x_1, \ldots, x_m) = \prod_{t=1}^n p(y_t|y_1 \ldots, y_{t-1}, c) \)
  
  – search: \( y^* = \arg \max_{y \in \mathcal{Y}} p(y|x) \)

• Representation learning: RNN, LSTM, CNN [Gehring et al., 2017]

• Search: greedy, beam search (no search at training time)

• Structured learning: [Ranzato et al., 2016; Wiseman and Rush, 2016]

• most recent: [Edunov et al., 2017]
Neural Network Models + Structured Perceptron-Inspired Updates

Watanabe and Sumita, 2015 uses a variant of Max Violation.
Neural Network Models + Structured Perceptron-Inspired Updates

Lee et al., 2016 extends Max Violation to All Violation.
Outline

- Three models for shift-reduce CCG parsing
  - representation learning:
  - structured learning:
  - search:
Outline

- Three models for shift-reduce CCG parsing
  - representation learning: struct. perceptron, Elman RNN, and LSTM
  - structured learning:
  - search:
Outline

• Three models for shift-reduce CCG parsing
  – representation learning: struct. perceptron, Elman RNN, and LSTM
  – structured learning: sequence-level training (global vs. local)
  – search:
Outline

• Three models for shift-reduce CCG parsing
  – representation learning: struct. perceptron, Elman RNN, and LSTM
  – structured learning: sequence-level training (global vs. local)
  – search: beam search for both training and testing
Outline

• Three models for shift-reduce CCG parsing
  – representation learning: struct. perceptron, Elman RNN, and LSTM
  – structured learning: sequence-level training (global vs. local)
  – search: beam search for both training and testing

from Heng et al., 2013
Outline

- Three models for shift-reduce CCG parsing
  - **representation learning**: struct. perceptron, Elman RNN, and LSTM
  - **structured learning**: sequence-level training (global vs. local)
  - **search**: beam search for both training and testing
Dependency Parsing

Parse me if you can.

Google SyntaxNet output
Transition-based Dependency Parsing

Configuration $c_i$

Action $c_i \rightarrow c_{i+1}$

Derivation $c_0, a_0 \rightarrow c_1, a_1 \rightarrow c_2, a_2$

source: Google SyntaxNet
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

the books which John likes
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

\[
\begin{align*}
\text{the} & \quad \frac{NP}{N} \\
\text{books} & \quad \frac{N}{N} \\
\text{which} & \quad \frac{(NP \setminus NP)/(S/NP)}{NP} \\
\text{John} & \quad \frac{NP}{NP} \\
\text{likes} & \quad \frac{(S \setminus NP)/NP}{NP}
\end{align*}
\]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

\[
\begin{array}{cccc}
\text{the} & \text{books} & \text{which} & \text{John} & \text{likes} \\
NP/N & N & (NP\backslash NP)/(S/\text{NP}) & NP & (S\backslash NP)/NP \\
\end{array}
\]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

```
the  books  which  John  likes
\[ \frac{NP/N}{N} \quad \frac{N}{(NP\setminus NP)/(S/\mathit{NP})} \quad \frac{NP}{\mathit{NP}} \quad \frac{(S\setminus \mathit{NP})/\mathit{NP}}{\mathit{NP}} \]
```
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

\[
\begin{align*}
\text{the} & \quad \text{books} & \quad \text{which} & \quad \text{John} & \quad \text{likes} \\
\text{NP} & \quad \text{N} & \quad (\text{NP} \backslash \text{NP}) \backslash (\text{S} \backslash \text{NP}) & \quad \text{NP} & \quad (\text{S} \backslash \text{NP}) \backslash \text{NP} \\
\text{NP} & \quad \rightarrow & \quad \text{NP} & \quad \rightarrow & \quad \text{T} \\
\text{S} & \quad \rightarrow & \quad \text{NP} & \quad \rightarrow & \quad \text{T}
\end{align*}
\]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

\[
\begin{align*}
\text{the} & \quad \text{books} & \quad \text{which} & \quad \text{John} & \quad \text{likes} \\
\frac{NP/N \quad N}{NP} & \quad \frac{(NP\backslash NP)/(S/NP)} & \quad \frac{NP}{(S\backslash NP)/NP} & \quad \frac{S/(S\backslash NP)}{S/NP} & \quad \frac{S/NP}{B}
\end{align*}
\]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

\[
\begin{align*}
\text{the} & \quad \text{books} \quad \text{which} \quad \text{John} \quad \text{likes} \\
NP/N & \quad N \quad (NP\backslash NP)/(S/NP) \quad NP \quad (S\backslash NP)/NP \\
\hline
\text{NP} & \quad (NP\backslash NP)/(S/NP) \quad S/(S\backslash NP) > T \\
\hline
S/NP & \quad S/NP > B \\
\hline
NP\backslash NP & \quad >
\end{align*}
\]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

```
NP/N  N  (NP\NP)/(S/NP)
   NP
```
```
NP
S/(S\NP)
   S/NP
```
```
NP\NP <
```
```
NP
```
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

- Parsing CCG
  - Supertagging (regular language; 1000 tags vs. 50 for CFG)
  - Parsing (mildly context-sensitive; only a dozen rules vs. 500K for CFG [Petrov and Klein, 2007])
Shift-Reduce CCG Parsing

• Combinatory Categorial Grammar (CCG)

• Parsing CCG
  – Supertagging (regular language; 1000 tags vs. 50 for CFG)
  – Parsing (mildly context-sensitive; only a dozen rules vs. 500K for CFG [Petrov and Klein, 2007])
Shift-Reduce CCG Parsing

• Combinatory Categorial Grammar (CCG)

• Parsing CCG – structured learning
  – Supertagging (regular language; 1000 tags vs. 50 for CFG)
  – Parsing (mildly context-sensitive; only a dozen rules vs. 500K for CFG [Petrov and Klein, 2007])
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

- Parsing CCG – structured learning
  - Supertagging (regular language; 1000 tags vs. 50 for CFG)
  - Parsing (mildly context-sensitive; only a dozen rules vs. 500K for CFG [Petrov and Klein, 2007])

- Dual Decomposition, Belief Propogation [Auli and Lopez, 2011]
Shift-Reduce CCG Parsing

- Combinatory Categorial Grammar (CCG)

- Parsing CCG – structured learning
  - Supertagging (regular language; 1000 tags vs. 50 for CFG)
  - Parsing (mildly context-sensitive; only a dozen rules vs. 500K for CFG [Petrov and Klein, 2007])

- Dual Decomposition, Belief Propogation [Auli and Lopez, 2011]

- Remains to be the most competitive formalism for recovering “deep” dependencies (from coordination, control, extraction etc.) [Rimell et al., 2009; Nivre et al., 2010]
Shift-Reduce CCG Parsing

the books which John likes
Shift-Reduce CCG Parsing

the books which John likes
Shift-Reduce CCG Parsing

the books which John likes
Shift-Reduce CCG Parsing

the books which John likes
Shift-Reduce CCG Parsing

the books which John likes

NP/N  N (NP\NP)/(S/NP)
Shift-Reduce CCG Parsing

the books which John likes

NP/N N (NP\NP)/(S/NP) NP

SH SH RE SH SH
Shift-Reduce CCG Parsing

the books which John likes
Shift-Reduce CCG Parsing

the books which John likes

NP/N  N  (NP/NP)/(S/NP)  NP  (S/NP)/NP

NP  NP  (S/(S/NP))  T
Shift-Reduce CCG Parsing

the books which John likes

NP/N N (NP\NP)/(S/NP)

NP

which

(S/NP)/NP

T

NP

S/(S/NP)

B

S/NP

SH SH RE SH SH UN SH RE
Shift-Reduce CCG Parsing

the books which John likes

NP/N N (NP\NP)/(S/\NP) NP (S/\NP)/NP

NP > T

S/(S/\NP) > B

NP/\NP

NP

SH SH RE SH SH SH UN SH RE RE RE
Model 1

[Xu et al., ACL 2014]
Standard Training: Greedy Local Model

the books which John likes

NP/N N (NP\NP)/(S/NP) NP (S/NP)/NP

SH SH RE SH SH UN SH RE RE RE
Standard Training: Greedy Local Model

- Score of an action $a = \mathbf{w} \cdot \phi(\langle s, q \rangle, a)$
- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack $(s_n, \ldots, s_1, s_0)$</th>
<th>queue $(q_0, q_1 \ldots, q_n)$</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
</tbody>
</table>
## Standard Training: Greedy Local Model

- Score of an action $a = w \cdot \phi(⟨s, q⟩, a)$
- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack ($s_n, \ldots, s_1, s_0$)</th>
<th>queue ($q_0, q_1 \ldots, q_n$)</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$N/N$</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
</tbody>
</table>
### Standard Training: Greedy Local Model

- Score of an action $a = w \cdot \phi(\langle s, q \rangle, a)$
- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack $(s_n, \ldots, s_1, s_0)$</th>
<th>queue $(q_0, q_1, \ldots, q_n)$</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$N/N$</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>$N/N N$</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
</tbody>
</table>
Standard Training: Greedy Local Model

- Score of an action \( a = w \cdot \phi(\langle s, q \rangle, a) \)
- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack ((s_n, \ldots, s_1, s_0))</th>
<th>queue ((q_0, q_1 \ldots, q_n))</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(N/N)</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>(N/N) (N)</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>3</td>
<td>(N)</td>
<td>plays Elianti</td>
<td>REDUCE</td>
</tr>
</tbody>
</table>
Standard Training: Greedy Local Model

- Score of an action $a = w \cdot \phi(\langle s, q \rangle, a)$
- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack $\langle s_n, \ldots, s_1, s_0 \rangle$</th>
<th>queue $\langle q_0, q_1, \ldots, q_n \rangle$</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$N/N$</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>$N/N \ N$</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>3</td>
<td>$N$</td>
<td>plays Elianti</td>
<td>REDUCE</td>
</tr>
<tr>
<td>4</td>
<td>$NP$</td>
<td>plays Elianti</td>
<td>UNARY</td>
</tr>
</tbody>
</table>
Standard Training: Greedy Local Model

- Score of an action $a = w \cdot \phi(\langle s, q \rangle, a)$

- No search at training time, can use beam search decoding

<table>
<thead>
<tr>
<th>step</th>
<th>stack $(s_n, \ldots, s_1, s_0)$</th>
<th>queue $(q_0, q_1 \ldots, q_n)$</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$N/N$</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>$N/N N$</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>3</td>
<td>$N$</td>
<td>plays Elianti</td>
<td>REDUCE</td>
</tr>
<tr>
<td>4</td>
<td>$NP$</td>
<td>plays Elianti</td>
<td>UNARY</td>
</tr>
<tr>
<td>5</td>
<td>$NP (S[dcl]\backslash NP)/NP$</td>
<td>Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>6</td>
<td>$NP (S[dcl]\backslash NP)/NP N$</td>
<td></td>
<td>SHIFT</td>
</tr>
<tr>
<td>7</td>
<td>$NP (S[dcl]\backslash NP)/NP NP$</td>
<td></td>
<td>UNARY</td>
</tr>
<tr>
<td>8</td>
<td>$NP S[dcl]\backslash NP$</td>
<td></td>
<td>REDUCE</td>
</tr>
<tr>
<td>9</td>
<td>$S[dcl]$</td>
<td></td>
<td>REDUCE</td>
</tr>
</tbody>
</table>
Global Structured Training
[Collins and Roark, 2004]

- Score of an action $a = w \cdot \phi(\langle s, q \rangle, a)$
Global Structured Training
[Collins and Roark, 2004]

- Score of an action $a = w \cdot \phi(\langle s, q \rangle, a)$
Global Structured Training
[Collins and Roark, 2004]

- Structured perceptron update: \( w \leftarrow w + \phi(x_i, y_{ij}) - \phi(x_i, B_j[0]) \)
Global Structured Training
[Collins and Roark, 2004]

- Structured perceptron update: \( \mathbf{w} \leftarrow \mathbf{w} + \phi(x_i, y_{ij}) - \phi(x_i, B_j[0]) \)
Global Structured Training
[Collins and Roark, 2004]

- Structured perceptron update: \( \mathbf{w} \leftarrow \mathbf{w} + \phi(x_i, y_{ij}) - \phi(x_i, B_j[0]) \)
Global Structured Training
[Collins and Roark, 2004]

- Structured perceptron update: \( \mathbf{w} \leftarrow \mathbf{w} + \phi(x_i, y_{ij}) - \phi(x_i, B_j[0]) \)
Global Structured Training for CCG

[Zhang and Clark, 2011]

- Conditional log-linear vs. linear
- Dynamic programming vs. beam search

<table>
<thead>
<tr>
<th>C&amp;C (Chart)</th>
<th>SR (Global)</th>
</tr>
</thead>
<tbody>
<tr>
<td>85.45</td>
<td>87.04</td>
</tr>
<tr>
<td>83.97</td>
<td>84.14</td>
</tr>
<tr>
<td>84.7</td>
<td>85.56</td>
</tr>
</tbody>
</table>
Spurious Ambiguity in CCG

\[
\begin{align*}
S/\langle S/\langle NPNP \rangle \rangle & \rightarrow \langle S/\langle NPNP \rangle \rangle \\
NPNP & \rightarrow \langle S/\langle NPNP \rangle \rangle \\
NP & \rightarrow \langle reads, book \rangle \\
N & \rightarrow \langle reads, he \rangle \\
S & \rightarrow \langle the, book \rangle \\
\end{align*}
\]

In general, exponentially many!
Motivation: Dependency Model

- The derivation is just a “trace” of the semantic interpretation [Steedman, 2000]
Motivation: Dependency Model

- The derivation is just a “trace” of the semantic interpretation [Steedman, 2000]
  - an elegant solution to the spurious ambiguity problem
  - gold-standard data cheaper to obtain
  - optimizing for evaluation
Model 1: The Dependency Model

- Use dependencies as the ground truth
  - encoding exponentially many “correct” paths

![Dependency Tree Example]

- A dependency oracle algorithm – online hypergraph search
- A learning algorithm adapting early update (under the violation-fixing struct. perceptron [Huang et al., 2012])
- Beam search – global structured learning
Model 1: The Dependency Model

- Use dependencies as the ground truth
  - encoding exponentially many “correct” paths
  - path selection is a hidden variable
Model 1: The Dependency Model

- Use dependencies as the ground truth
  - encoding exponentially many "correct" paths
  - path selection is a hidden variable

- A dependency oracle algorithm – online hypergraph search
Model 1: The Dependency Model

• Use dependencies as the ground truth
  – encoding exponentially many “correct” paths
  – path selection is a hidden variable

• A dependency oracle algorithm – online hypergraph search

• A learning algorithm adapting early update (under the violation-fixing struct. perceptron [Huang et al., 2012])
Model 1: The Dependency Model

• Use dependencies as the ground truth
  – encoding exponentially many “correct” paths
  – path selection is a hidden variable

• A dependency oracle algorithm – online hypergraph search

• A learning algorithm adapting early update (under the violation-fixing struct. perceptron [Huang et al., 2012])

• Beam search – global structured learning
## The Dependency Model

<table>
<thead>
<tr>
<th></th>
<th>[Clark et al., 2002]</th>
<th>C&amp;C (dep)</th>
<th>Z&amp;C</th>
<th>this work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shift-Reduce</td>
<td>✗</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Dep. Model</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Deriv. Feats</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
CCG Parse Forest

- Compactly represents all derivation and dependency structure pair
- Grouping together equivalent chart entries
  - identical *category*, *head* and *unfilled* dependencies
  - individual entries are *conjunctive* nodes and equivalence classes are *disjunctive* nodes
CCG Parse Forest

- Compactly represents all derivation and dependency structure pair
- Grouping together equivalent chart entries
  - identical category, head and unfilled dependencies
  - individual entries are conjunctive nodes and equivalence classes are disjunctive nodes
The Oracle Forest

- A *subset* of the *complete* forest
  - consistent with the gold-standard dependency structure
  - exponentially-sized and impossible to enumerate

- A dependency structure decomposes over derivations
  - dependencies are realized on conjunctive nodes
  - can count dependencies on-the-fly
The Oracle Forest

- Intuition 1: Dependencies “live on” conjunctive nodes
The Oracle Forest

- intuition 1: dependencies “live on” conjunctive nodes

\[ \langle \text{some, } NP/N_1, 1, \text{ books} \rangle \]
The Oracle Forest

• intuition 1: dependencies “live on” conjunctive nodes

⟨some, NP/N₁, 1, books⟩
⟨bought, (S\NP₁)/NP₂, 2, books⟩
The Oracle Forest

• intuition 1: dependencies “live on” conjunctive nodes

\[ \langle \text{some}, \text{NP}/N_1, 1, \text{books} \rangle \]
\[ \langle \text{bought}, (S\setminus NP_1)/NP_2, 2, \text{books} \rangle \]
\[ \langle \text{he}, (S\setminus NP_1)/NP_2, 1, \text{bought} \rangle \]
The Oracle Forest

• intuition 1: dependencies “live on” conjunctive nodes

\[
\langle \text{some}, \ NP/N_1, 1, \ books \rangle
\]

\[
\langle \text{bought}, \ (S\backslash NP_1)/NP_2, 2, \ books \rangle
\]

\[
\langle \text{he}, \ (S\backslash NP_1)/NP_2, 1, \ bought \rangle
\]
The Oracle Forest

- intuition 1: dependencies “live on” conjunctive nodes
- intuition 2: a conj. node that has less than the max possible number of gold-standard dependencies is not gold (optimal substructure)

\[
\langle \text{some, } NP/N_1, 1, \text{ books} \rangle \\
\langle \text{bought, } (S\backslash NP_1)/NP_2, 2, \text{ books} \rangle \\
\langle \text{he, } (S\backslash NP_1)/NP_2, 1, \text{ bought} \rangle
\]
The Oracle Forest

- Intuition 1: dependencies “live on” conjunctive nodes
- Intuition 2: a conj. node that has less than the max possible number of gold-standard dependencies is not gold (optimal substructure)
The Oracle Forest

- Intuition 1: dependencies “live on” conjunctive nodes
- Intuition 2: a conj. node that has less than the max possible number of gold-standard dependencies is not gold (optimal substructure)

\[
\langle \text{some, } NP/N_1, 1, \text{ books} \rangle \\
\langle \text{bought, } (S\backslash NP_1)/NP_2, 2, \text{ books} \rangle \\
\langle \text{he, } (S\backslash NP_1)/NP_2, 1, \text{ bought} \rangle
\]
Shift-Reduce Dependency Oracle

- The dependency oracle

\[ f_d(\langle s, q \rangle, (x, c), \Phi_G) = \begin{cases} 
  \text{true} & \text{if } s' \sim G \text{ or } s' \simeq G \\
  \text{false} & \text{otherwise}
\end{cases} \]
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

```
S[dcl]
  NP
    N
      N/N
        Mr. President
    (S[dcl]\NP)/NP
      visited
        N
          Paris
  S[dcl]\NP
```
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

\[
\begin{array}{c}
\text{S[dcl]} \\
\text{NP} \\
\text{N} \\
\text{N/N} \\
\text{Mr.} \\
\text{President}
\end{array}
\quad \begin{array}{c}
\text{S[dcl]\NP} \\
\text{(S[dcl]\NP)/NP} \\
\text{visited} \\
\text{N} \\
\text{Paris}
\end{array}
\]

Shift Shift
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

Shift Shift Reduce
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

Shift Shift Reduce Unary
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

Shift Shift Reduce Unary Shift
The Dependency Model Oracle

Canonical Shift-Reduce is bottom-up post-order traversal

Shift Shift Reduce Unary Shift Shift Unary Reduce Reduce
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest

He some books

Shift-NP Shift-(S\NP)/NP
But this doesn’t carry over to an oracle forest

Shift-\(NP\)  Shift-\((S\backslash NP)/NP\)
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest

Shift-NP  Shift-(S\NP)/NP  Shift-NP/N
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest

[Diagram of dependency tree with labeled nodes: He, bought, some, books.]

Shift-NP  Shift-(S\NP)/NP  Shift-NP/N
But this doesn’t carry over to an oracle forest
The Dependency Model Oracle

But this doesn’t carry over to an oracle forest

Shift-NP  Shift-(S\NP)/NP  Shift-NP/N  Shift-N
The Dependency Model Oracle
The Dependency Model Oracle

Mr. President visited Paris

\[ \text{S[dcl]} \]

\[ \text{NP} \]

\[ \text{N} \quad (\text{S[dcl]}\backslash\text{NP})/\text{NP} \]

\[ \text{NP} \quad \text{visited} \quad \text{N} \]

\[ \text{N/N} \quad \text{N} \]

Mr. President

\[ (\text{S[dcl]}\backslash\text{NP})/\text{NP} \]
The Dependency Model Oracle

- The dependency oracle

\[ f_d(⟨s, q⟩, (x, c), Φ_G) = \begin{cases} 
true & \text{if } s' \sim G \text{ or } s' \simeq G \\
false & \text{otherwise}
\end{cases} \]

- Shared ancestor set
  - contains possible valid nodes an item should visit
  - is built on-the-fly during decoding for each action type
  - constructed with each valid action
The Dependency Model

He some books

stack \( (s_n, \ldots, s_1, s_0) \) \( \mathcal{R}(c_{s_0}) \)
The Dependency Model

He bought some books

stack \((s_n, \ldots, s_1, s_0)\) \(R(c_{s_0})\)
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\)  \(\mathcal{R}(c_{s_0})\)

\(NP\)  

\(()\)
The Dependency Model

stack \( (s_n, \ldots, s_1, s_0) \) \[ \text{R}(c_{s_0}) \]

\( NP \)

\( NP (S \backslash NP) / NP \)
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\) \hspace{1cm} \mathcal{R}(c_{s_0})

\rule{0.5cm}{0.5pt}

\(\text{NP} \) \hspace{1cm} ()

\(\text{NP} (S \backslash \text{NP})/\text{NP} \) \hspace{1cm} (S, S)
The Dependency Model

He bought some books

\[ \text{stack} \left( s_n, \ldots, s_1, s_0 \right) \]
\[ R(c_{s_0}) \]

\[ NP \]
\[ NP (S\backslash NP)/NP \]
\[ NP (S\backslash NP)/NP NP/N \]
The Dependency Model

He bought some books

stack \( (s_n, \ldots, s_1, s_0) \)

\( R(c_{s_0}) \)

NP

\( NP \)

\( NP \ (S\ NP) / NP \)

\( NP \ (S\ NP) / NP \ NP / N \)
The Dependency Model

He bought some books

stack \((s_n, \ldots, s_1, s_0)\)  \(R(c_{s_0})\)

<table>
<thead>
<tr>
<th>NP</th>
<th>()</th>
</tr>
</thead>
<tbody>
<tr>
<td>NP (S\NP)/NP</td>
<td>(S, S)</td>
</tr>
<tr>
<td>NP (S\NP)/NP NP/N</td>
<td>(S\NP, (S\NP)/N)</td>
</tr>
</tbody>
</table>
The Dependency Model

```
stack (s_n, ..., s_1, s_0)  |  R(c_{s_0})
NP                        | ()
NP (S\NP)/NP             | (S, S)
NP (S\NP)/NP NP/N        | (S\NP, (S\NP)/N)
NP (S\NP)/NP NP/N N      |
```
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\) \hspace{1cm} \mathcal{R}(c_{s_0})

\begin{align*}
NP & \hspace{1cm} () \\
NP (S\backslash NP) / NP & \hspace{1cm} (S, S) \\
NP (S\backslash NP) / NP NP / N & \hspace{1cm} (S\backslash NP, (S\backslash NP) / N) \\
NP (S\backslash NP) / NP NP / N N & \hspace{1cm} \text{(}\hspace{1cm})
\end{align*}
The Dependency Model

```
stack (s_n, ..., s_1, s_0)  \[R(c_{s_0})\]
-----------------------------
NP                             ()
NP (S\NP)/NP                  (S, S)
NP (S\NP)/NP NP/N             (S\NP, (S\NP)/N)
NP (S\NP)/NP NP/N N           (NP)
```
The Dependency Model

He bought some books

stack \( (s_n, \ldots, s_1, s_0) \) \( \mathcal{R}(c_{s_0}) \)

\begin{align*}
\text{NP} & \quad () \\
\text{NP} (S\backslash NP)/NP & \quad (S, S) \\
\text{NP} (S\backslash NP)/NP \text{ NP}/N & \quad (S\backslash NP, (S\backslash NP)/N) \\
\text{NP} (S\backslash NP)/NP \text{ NP}/N \text{ N} & \quad (NP) \\
\text{NP} (S\backslash NP)/NP & \quad ()
\end{align*}
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\) | \(R(c_{s_0})\)
--- | ---
\(NP\) | ()
\(NP (S\setminus NP)/NP\) | \((S, S)\)
\(NP (S\setminus NP)/NP NP/N\) | \((S\setminus NP, (S\setminus NP)/N)\)
\(NP (S\setminus NP)/NP NP/N N\) | \((NP)\)
The Dependency Model

```

```

<table>
<thead>
<tr>
<th>stack ((s_n, \ldots, s_1, s_0))</th>
<th>(\mathcal{R}(c_{s_0}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>(NP)</td>
<td>(())</td>
</tr>
<tr>
<td>(NP \ (S \backslash NP)/NP)</td>
<td>((S, S))</td>
</tr>
<tr>
<td>(NP \ (S \backslash NP)/NP \ NP/N)</td>
<td>((S \backslash NP, (S \backslash NP)/N))</td>
</tr>
<tr>
<td>(NP \ (S \backslash NP)/NP \ NP/N \ N)</td>
<td>((NP))</td>
</tr>
<tr>
<td>(NP \ (S \backslash NP)/NP \ NP)</td>
<td>((S \backslash NP))</td>
</tr>
</tbody>
</table>

```
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\) \quad \mathcal{R}(c_{s_0})

\begin{align*}
NP & (\ ) \\
NP (S\ NP)/NP & (S, S) \\
NP (S\ NP)/NP NP/N & (S\ NP, (S\ NP)/N) \\
NP (S\ NP)/NP NP/N N & (NP) \\
NP (S\ NP)/NP NP & (S\ NP) \\
NP S\ NP
\end{align*}
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\) | \(\mathcal{R}(c_{s_0})\)
\hline
NP & ()
NP \((S\backslash NP)/NP\) & \((S, S)\)
NP \((S\backslash NP)/NP\) \(NP/N\) & \((S\backslash NP, (S\backslash NP)/N)\)
NP \((S\backslash NP)/NP\) \(NP/N\) \(N\) & \((NP)\)
NP \((S\backslash NP)/NP\) \(NP\) & \((S\backslash NP)\)
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\)

\(\mathcal{R}(c_{s_0})\)

\(NP\) 
()-

\(NP (S\setminus NP)/NP\) 
\((S, S)\)

\(NP (S\setminus NP)/NP\) 
\((S\setminus NP, (S\setminus NP)/N)\)

\(NP (S\setminus NP)/NP\) 
\((NP)\)

\(NP (S\setminus NP)/NP\) 
\((S\setminus NP)\)

\(NP S\setminus NP\) 
\((S)\)
The Dependency Model

stack \((s_n, \ldots, s_1, s_0)\)

\(\mathcal{R}(c_{s_0})\)

<table>
<thead>
<tr>
<th>(NP)</th>
<th>()</th>
</tr>
</thead>
<tbody>
<tr>
<td>(NP) ((S\setminus NP)/NP)</td>
<td>((S, S))</td>
</tr>
<tr>
<td>(NP) ((S\setminus NP)/NP) (NP/N)</td>
<td>((S\setminus NP, (S\setminus NP)/N))</td>
</tr>
<tr>
<td>(NP) ((S\setminus NP)/NP) (NP/N) (N)</td>
<td>((NP))</td>
</tr>
<tr>
<td>(NP) ((S\setminus NP)/NP) (NP)</td>
<td>((S\setminus NP))</td>
</tr>
<tr>
<td>(NP) ((S\setminus NP))</td>
<td>((S))</td>
</tr>
<tr>
<td>(S)</td>
<td>()</td>
</tr>
</tbody>
</table>
Training: Chart-based Dependency Model

- Exponentially many derivations $\omega$ consistent with a dependency structure $\pi$ [Clark and Curran, 2007]

\[
P(\pi|S) = \sum_{\omega \in \Delta(\pi)} P(\omega, \pi|S)
\]
Training: Chart-based Dependency Model

- Exponentially many derivations $\omega$ consistent with a dependency structure $\pi$ [Clark and Curran, 2007]

$$P(\pi | S) = \sum_{\omega \in \Delta(\pi)} P(\omega, \pi | S)$$

$$L'(\Lambda) = L(\Lambda) - G(\Lambda)$$

$$= \log \prod_{j=1}^{m} P_{\Lambda}(\pi_j | S_j) - \sum_{i=1}^{n} \frac{\lambda_i^2}{2\sigma_i^2}$$

$$= \sum_{j=1}^{m} \log \frac{\sum_{d \in \Delta(\pi_j)} e^{\lambda \cdot f(d, \pi_j)}}{\sum_{\omega \in \rho(S_j)} e^{\lambda \cdot f(\omega)}} - \sum_{i=1}^{n} \frac{\lambda_i^2}{2\sigma_i^2}$$
Training: Chart-based Dependency Model

- Exponentially many derivations $\omega$ consistent with a dependency structure $\pi$ [Clark and Curran, 2007]

\[
P(\pi|S) = \sum_{\omega \in \Delta(\pi)} P(\omega, \pi|S)
\]

\[
L'(\Lambda) = L(\Lambda) - G(\Lambda)
\]

\[
= \log \prod_{j=1}^{m} P_\Lambda(\pi_j|S_j) - \sum_{i=1}^{n} \frac{\chi_i^2}{2\sigma_i^2}
\]

\[
= \sum_{j=1}^{m} \log \frac{\sum_{d \in \Delta(\pi_j)} e^{\lambda \cdot f(d, \pi_j)}}{\sum_{\omega \in \rho(S_j)} e^{\lambda \cdot f(\omega)}} - \sum_{i=1}^{n} \frac{\lambda_i^2}{2\sigma_i^2}
\]

- Requires summing over all $\omega$
Online Training

• The normal-form model uses the perceptron with early update
  
  – only one correct sequence
  
  – “violation” is guaranteed [Huang et al., 2012]
Online Training

• Standard early update no longer valid for the dependency model
  – multiple correct items possible in each beam
  – “violation” is not guaranteed [Huang et al, 2012]
Online Training

- Standard early update no longer valid for the dependency model
  - multiple correct items possible in each beam
  - “violation” is not guaranteed [Huang et al, 2012]
  - $\mathbf{w} \leftarrow \mathbf{w} + \phi(\Pi_G[0]) - \phi(B_i[0])$

from Heng et al., 2013
Results

<table>
<thead>
<tr>
<th>C&amp;C (Hybrid)</th>
<th>SR (Normal)</th>
<th>SR (Dep)</th>
</tr>
</thead>
<tbody>
<tr>
<td>86.24</td>
<td>87.43</td>
<td>87.03</td>
</tr>
<tr>
<td>84.17</td>
<td>83.61</td>
<td>85.08</td>
</tr>
<tr>
<td>85.19</td>
<td>85.48</td>
<td>86.04</td>
</tr>
</tbody>
</table>
Results

Recall %
Dependency length (bins of 5)
sr-dep
sr-normal
Model 2

[Xu et al., NAACL 2016]
Shift-Reduce Parsing

- Linear model (struct. perceptron, SVM etc.)
  
  \[- \text{score}(y_i) = w \cdot \phi(\langle s, q, y_i \rangle) \]
  \[- \text{score}(y) = \sum_{i=1}^{\|y\|} \text{score}(y_i) \]
  \[- y^* = \arg \max_{y \in \mathcal{Y}_x} \text{score}(y) \]
Shift-Reduce Parsing

• Linear model (struct. perceptron, SVM etc.)

  – $score(y_i) = w \cdot \phi(⟨s, q⟩, y_i)$
  
  – $score(y) = \sum_{i=1}^{\left|y\right|} score(y_i)$

  – $y^* = \arg \max_{y \in \mathcal{Y}_x} score(y)$

• Great flexibility in defining the feature functions
Shift-Reduce Parsing

• Linear model (struct. perceptron, SVM etc.)

  – \( score(y_i) = \mathbf{w} \cdot \phi(\langle s, q \rangle, y_i) \)

  – \( score(y) = \sum_{i=1}^{\vert y \vert} score(y_i) \)

  – \( y^* = \arg \max_{y \in \mathcal{Y}_x} score(y) \)

• Great flexibility in defining the feature functions

  – results in millions of indicator features
Shift-Reduce Parsing

• Linear model (struct. perceptron, SVM etc.)
  
  – \( \text{score}(y_i) = w \cdot \phi(\langle s, q \rangle, y_i) \)
  
  – \( \text{score}(y) = \sum_{i=1}^{\|y\|} \text{score}(y_i) \)
  
  – \( y^* = \arg \max_{y \in \mathcal{Y}_x} \text{score}(y) \)

• Great flexibility in defining the feature functions
  
  – results in millions of indicator features
  
  – sparse and expensive to compute
Shift-Reduce Parsing

• Linear model (struct. perceptron, SVM etc.)
  
  \[\text{score}(y_i) = w \cdot \phi(\langle s, q \rangle, y_i)\]
  
  \[\text{score}(y) = \sum_{i=1}^{\lfloor y \rfloor} \text{score}(y_i)\]
  
  \[y^* = \arg \max_{y \in Y_x} \text{score}(y)\]

• Great flexibility in defining the feature functions
  
  – results in millions of indicator features
  
  – sparse and expensive to compute

• [Yamada and Matsumoto, 2003; Huang and Sagae, 2010; Zhang and Clark, 2011; Zhang and Nivre, 2011; Goldberg and Nivre, 2012; Bohnet et al., 2013; Zhu et al., 2013]
Sparse Features

<table>
<thead>
<tr>
<th>feature templates</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. $S_{0wp}, S_{0c}, S_{0pc}, S_{0wc},$</td>
</tr>
<tr>
<td>$S_{1wp}, S_{1c}, S_{1pc}, S_{1wc},$</td>
</tr>
<tr>
<td>$S_{2pc}, S_{2wc},$</td>
</tr>
<tr>
<td>$S_{3pc}, S_{3wc},$</td>
</tr>
<tr>
<td>2. $Q_{0wp}, Q_{1wp}, Q_{2wp}, Q_{3wp},$</td>
</tr>
<tr>
<td>3. $S_{0Lpc}, S_{0Lwc}, S_{0Rpc}, S_{0Rwc},$</td>
</tr>
<tr>
<td>$S_{0Upc}, S_{0Uwc},$</td>
</tr>
<tr>
<td>$S_{1Lpc}, S_{1Lwc}, S_{1Rpc}, S_{1Rwc},$</td>
</tr>
<tr>
<td>$S_{1Upc}, S_{1Uwc},$</td>
</tr>
<tr>
<td>4. $S_{0wcS_{1wc}}, S_{0cS_{1w}}, S_{0wS_{1c}}, S_{0cS_{1c}},$</td>
</tr>
<tr>
<td>$S_{0wcQ_{0wp}}, S_{0cQ_{0wp}}, S_{0wcQ_{0p}}, S_{0cQ_{0p}},$</td>
</tr>
<tr>
<td>$S_{1wcQ_{0wp}}, S_{1cQ_{0wp}}, S_{1wcQ_{0p}}, S_{1cQ_{0p}},$</td>
</tr>
<tr>
<td>5. $S_{0wcS_{1cQ_{0p}}}, S_{0cS_{1wcQ_{0p}}}, S_{0cS_{1cQ_{0wp}}},$</td>
</tr>
<tr>
<td>$S_{0cS_{1cQ_{0p}}}, S_{0pS_{1pQ_{0p}}},$</td>
</tr>
<tr>
<td>$S_{0wcQ_{0pQ_{1p}}}, S_{0cQ_{0wpQ_{1p}}}, S_{0cQ_{0pQ_{1wp}}},$</td>
</tr>
<tr>
<td>$S_{0cQ_{0pQ_{1p}}}, S_{0pQ_{0pQ_{1p}}},$</td>
</tr>
<tr>
<td>$S_{0wcS_{1cS_{2c}}}, S_{0cS_{1wcS_{2c}}}, S_{0cS_{1cS_{2wc}}},$</td>
</tr>
<tr>
<td>$S_{0cS_{1cS_{2c}}}, S_{0pS_{1pS_{2p}}},$</td>
</tr>
<tr>
<td>6. $S_{0cS_{0HeS_{0Lc}}}, S_{0cS_{0HeS_{0Rc}}},$</td>
</tr>
<tr>
<td>$S_{1cS_{1HeS_{1Rc}}},$</td>
</tr>
<tr>
<td>$S_{0cS_{0RcQ_{0p}}}, S_{0cS_{0RcQ_{0w}}},$</td>
</tr>
<tr>
<td>$S_{0cS_{0LcS_{1c}}}, S_{0cS_{0LcS_{1w}}},$</td>
</tr>
<tr>
<td>$S_{0cS_{1cS_{1Rc}}, S_{0wS_{1cS_{1Rc}}},}$</td>
</tr>
</tbody>
</table>

Table 1: Feature templates.

[Zhang and Clark, 2011]
Kernel Features [Chen and Manning, 2014]

<table>
<thead>
<tr>
<th>$s_0.w$</th>
<th>$s_1.w$</th>
<th>$s_2.w$</th>
<th>$s_3.w$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s.w_0$</td>
<td>$s.w_1$</td>
<td>$s.w_2$</td>
<td>$s.w_3$</td>
</tr>
<tr>
<td>$s_0.l.w$</td>
<td>$s_1.l.w$</td>
<td>$s_0.r.w$</td>
<td>$s_1.r.w$</td>
</tr>
<tr>
<td>$q_0.w$</td>
<td>$q_1.w$</td>
<td>$q_2.w$</td>
<td>$q_3.w$</td>
</tr>
<tr>
<td>$s_0.c$</td>
<td>$s_0.l.c$</td>
<td>$s_0.r.c$</td>
<td></td>
</tr>
<tr>
<td>$s_1.c$</td>
<td>$s_1.l.c$</td>
<td>$s_1.r.c$</td>
<td></td>
</tr>
<tr>
<td>$s_2.c$</td>
<td>$s_3.c$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Kernel Features [Chen and Manning, 2014]

Softmax layer:
\[ p = \text{softmax}(W_2h) \]

Hidden layer:
\[ h = (W_1^w x^w + W_1^t x^t + W_1^l x^l + b_1)^3 \]

Input layer: \([x^w, x^t, x^l]\)

Configuration

Stack

Buffer

words

POS tags

arc labels

ROOT has_VBZ good JJ

control NN ...

He PRP

nsbj
Kernel Features \cite{Chen and Manning, 2014}

State-of-the-art results at the time!
Local Normalization

$$p(y_t | \langle s, q \rangle_y^{t-1}; \theta) = \frac{\exp\{\gamma(y_t, \langle s, q \rangle_y^{t-1}; \theta)\}}{Z_L (\langle s, q \rangle_y^{t-1})}$$
Local Normalization

\[ p(y_t | \langle s, q \rangle_{y}^{t-1}; \theta) = \frac{\exp\{\gamma(y_t, \langle s, q \rangle_{y}^{t-1}; \theta)\}}{Z_L (\langle s, q \rangle_{y}^{t-1})} \]

\[ Z_L (\langle \alpha, \beta \rangle_{y}^{t-1}) = \sum_{y_t' \in \mathcal{T}(\langle \alpha, \beta \rangle_{y}^{t-1})} \exp\{\gamma(y_t', \langle \alpha, \beta \rangle_{y}^{t-1}; \theta)\} \]
Local Normalization

\[ p(y_t|\langle s, q\rangle^t_y; \theta) = \frac{\exp\{\gamma(y_t, \langle s, q\rangle^t_y; \theta)\}}{Z_L(\langle s, q\rangle^t_y)} \]

\[ Z_L(\langle \alpha, \beta\rangle^t_y) = \sum_{y_t' \in \mathcal{T}(\langle \alpha, \beta\rangle^t_y)} \exp\{\gamma(y_t', \langle \alpha, \beta\rangle^t_y; \theta)\} \]

\[ p(y|\theta) = \prod_{t=1}^{\vert y \vert} p(y_t|\langle \alpha, \beta\rangle^t_y; \theta) = \frac{\exp\{\sum_{t=1}^{\vert y \vert} \gamma(y_t, \langle \alpha, \beta\rangle^t_y; \theta)\}}{\prod_{t=1}^{\vert y \vert} Z_L(\langle \alpha, \beta\rangle^t_y)} \]
Global Normalization (CRF)

\[ p(y|\theta) = \frac{\exp\{\sum_{t=1}^{\mid y \mid} \gamma(y_t, \langle \alpha, \beta \rangle_{y, t-1}; \theta)\}}{Z_G} \]
Global Normalization (CRF)

\[
p(y|\theta) = \frac{\exp\left\{\sum_{t=1}^{|y|} \gamma(y_t, \langle\alpha, \beta\rangle_{y}^{t-1}; \theta)\right\}}{Z_G}
\]

\[
Z_G = \sum_{y' \in S_{|y|}} \exp \sum_{t=1}^{|y|} \gamma(y'_t, \langle\alpha, \beta\rangle_{y'}^{t-1}; \theta)
\]
Global Normalization (CRF)

\[
p(y|\theta) = \frac{\exp\left\{\sum_{t=1}^{\left|y\right|} \gamma(y_t, \langle \alpha, \beta \rangle_{y_t}^{t-1}; \theta)\right\}}{Z_G}
\]

\[
Z_G = \sum_{y' \in S_{\left|y\right|}} \exp \sum_{t=1}^{\left|y\right|} \gamma(y'_t, \langle \alpha, \beta \rangle_{y'_t}^{t-1}; \theta)
\]

\[
y^* = \arg \max_{y' \in S_{\left|y\right|}} \sum_{t=1}^{\left|y\right|} \gamma(y'_t, \langle \alpha, \beta \rangle_{y'_t}^{t-1}; \theta)
\]

[Zhou et al., 2015; Andor et al., 2016]
Local vs. Global Normalization

\[ Z_L((\langle \alpha, \beta \rangle_y)^{t-1}) = \sum_{y_t' \in T((\langle \alpha, \beta \rangle_y)^{t-1})} \exp\{\gamma(y_t', (\langle \alpha, \beta \rangle_y)^{t-1}; \theta)\} \]

\[ Z_G = \sum_{y' \in S_{|y|}} \exp \sum_{t=1}^{|y|} \gamma(y_t', (\langle \alpha, \beta \rangle_{y'})^{t-1}; \theta) \]

The label bias problem [Bottou et al., 1997; Le Cun et al., 1998; Lafferty et al., 2001]; Andor et al., 2016 showed that \( \mathcal{P}_L \subset \mathcal{P}_G \) (assuming no lookahead)
Expected F-measure Training for Shift-Reduce Parsing with RNNs

<table>
<thead>
<tr>
<th>C&amp;M, 2014</th>
<th>NN</th>
<th>Beam (Train)</th>
<th>Beam (Test)</th>
<th>global</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
</tbody>
</table>
Expected F-measure Training for Shift-Reduce Parsing with RNNs

<table>
<thead>
<tr>
<th>Model</th>
<th>NN</th>
<th>Beam (Train)</th>
<th>Beam (Test)</th>
<th>global</th>
</tr>
</thead>
<tbody>
<tr>
<td>C&amp;M, 2014</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
<td>✗</td>
</tr>
<tr>
<td>present model</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>
## Expected F-measure Training for Shift-Reduce Parsing with RNNs

<table>
<thead>
<tr>
<th></th>
<th>NN</th>
<th>Beam (Train)</th>
<th>Beam (Test)</th>
<th>global</th>
</tr>
</thead>
<tbody>
<tr>
<td>C&amp;M, 2014</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>present model</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

At the same time, the model is optimized towards the final evaluation metric ✓
Expected F-Measure Training: Step 1

1. Train a baseline model using a cross-entropy loss (pretraining)
Expected F-Measure Training: Step 1

1. Train a baseline model using a cross-entropy loss (pretraining)
Expected F-Measure Training: Step 1

1. Train a baseline model using a cross-entropy loss (pretraining)
Expected F-Measure Training: Step 1

1. Train a baseline model using a cross-entropy loss (pretraining)

\[ L(\theta_B) = - \sum_{k}^{T_i} p(t_k|\theta_B), \quad \theta_B = \{U, V, W\} \]
2 Let $\theta = \theta_B$, and parse each sentence in the training data with beam search
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.

$$\gamma(y_i) = |y_i| \sum_{j=1} \log s_\theta(y_{ij})$$
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.

$$
\gamma(y_i) = \sum_{j=1}^{\vert y_i \vert} \log s_\theta(y_{ij})
$$
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.

\[
\gamma(y_i) = \sum_{j=1}^{\vert y_i \vert} \log s_\theta(y_{ij}), \quad F_1(\Delta y_i, \Delta^G_{x_n})
\]
Let $\theta = \theta_B$, and parse each sentence in the training data with beam search.

$$\gamma(y_i) = \sum_{j=1}^{|y_i|} \log s_\theta(y_{ij}), \quad F_1(\Delta y_i, \Delta x_n^G)$$
Expected F-Measure Training: Step 3

3. Compute the -XF1 loss for each sentence, do SGD update and iterate

\[ J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i | \theta) F1(\Delta y_i, \Delta^G_{x_n}), \]

\[ p(y_i | \theta) = \frac{\exp\{\gamma(y_i)\}}{\sum_{y \in \Lambda(x_n)} \exp\{\gamma(y)\}} \]
Expected F-Measure Training: Step 3

3. Compute the -XF1 loss for each sentence, do SGD update and iterate

\[ J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i|\theta) F1(\Delta y_i, \Delta^G_{x_n}), \]

\[ p(y_i|\theta) = \frac{\exp\{\gamma(y_i)\}}{\sum_{y \in \Lambda(x_n)} \exp\{\gamma(y)\}} \]
Expected F-Measure Training: Step 3

3 Compute the -XF1 loss for each sentence, do SGD update and iterate

$$J(\theta) = -\text{XF1}(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i|\theta) F1(\Delta y_i, \Delta^G_{x_n}),$$

$$p(y_i|\theta) = \frac{\exp\{\gamma(y_i)\}}{\sum_{y \in \Lambda(x_n)} \exp\{\gamma(y)\}}$$

$$\frac{\partial J(\theta)}{\partial \theta} = - \sum_{y_i \in \Lambda(x_n)} \sum_{y_{ij} \in y_i} \frac{\partial J(\theta)}{\partial s_\theta(y_{ij})} \frac{\partial s_\theta(y_{ij})}{\partial \theta}$$
Expected F-Measure Training: Step 3

3 Compute the -$XF1$ loss for each sentence, do SGD update and iterate

\[ J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i|\theta) F1(\Delta y_i, \Delta x_n), \]

\[ p(y_i|\theta) = \frac{\exp\{\gamma(y_i)\}}{\sum_{y \in \Lambda(x_n)} \exp\{\gamma(y)\}} \]

\[ \frac{\partial J(\theta)}{\partial \theta} = - \sum_{y_i \in \Lambda(x_n)} \sum_{y_{ij} \in y_i} \frac{\partial J(\theta)}{\partial s_\theta(y_{ij})} \frac{\partial s_\theta(y_{ij})}{\partial \theta} \]
Compute the -XF1 loss for each sentence, do SGD update and iterate

\[
\frac{\partial J(\theta)}{\partial \theta} = - \sum_{y_i \in \Lambda(x_n)} \sum_{y_{ij} \in y_i} \frac{\partial J(\theta)}{\partial s_{\theta}(y_{ij})} \frac{\partial s_{\theta}(y_{ij})}{\partial \theta}
\]
Compute the -$XF1$ loss for each sentence, do SGD update and iterate

\[
\frac{\partial J(\theta)}{\partial \theta} = - \sum_{y_i \in \Lambda(x_n)} \sum_{y_{ij} \in y_i} \frac{\partial J(\theta)}{\partial s_\theta(y_{ij})} \frac{\partial s_\theta(y_{ij})}{\partial \theta}
\]
## Expected F-Measure Training

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>$\gamma(y_i)$</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>$y_{11} \ y_{12} \ldots \ y_{1i}$</td>
<td>-0.60</td>
<td>0.67</td>
</tr>
<tr>
<td>$y_2$</td>
<td>$y_{21} \ y_{22} \ldots \ y_{2j}$</td>
<td>-1.5</td>
<td>0.81</td>
</tr>
<tr>
<td>$y_3$</td>
<td>$y_{31} \ y_{32} \ldots \ y_{3k}$</td>
<td>-4.96</td>
<td>0.90</td>
</tr>
</tbody>
</table>
# Expected F-Measure Training

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>$\gamma(y_i)$</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>$y_{11} \ y_{12} \ldots \ y_{1i}$</td>
<td>-0.60</td>
<td>0.67</td>
</tr>
<tr>
<td>$y_2$</td>
<td>$y_{21} \ y_{22} \ldots \ y_{2j}$</td>
<td>-1.5</td>
<td>0.81</td>
</tr>
<tr>
<td>$y_3$</td>
<td>$y_{31} \ y_{32} \ldots \ y_{3k}$</td>
<td>-4.96</td>
<td>0.90</td>
</tr>
</tbody>
</table>

$$J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i | \theta)F1(\Delta y_i, \Delta y_{x_n}) = -71.00$$
## Expected F-Measure Training

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>( \gamma(y_i) )</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( y_1 )</td>
<td>( y_{11} y_{12} \ldots y_{1i} )</td>
<td>-0.60</td>
<td>0.67</td>
</tr>
<tr>
<td>( y_2 )</td>
<td>( y_{21} y_{22} \ldots y_{2j} )</td>
<td>-1.50</td>
<td>0.81</td>
</tr>
<tr>
<td>( y_3 )</td>
<td>( y_{31} y_{32} \ldots y_{3k} )</td>
<td>-4.96</td>
<td>0.90</td>
</tr>
</tbody>
</table>

\[
J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i|\theta)F1(\Delta y_i, \Delta^G_{x_n}) = -71.00
\]

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>( \gamma(z_i) )</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>( z_1 )</td>
<td>( z_{11} z_{12} \ldots z_{1i} )</td>
<td>-0.90</td>
<td>0.71</td>
</tr>
<tr>
<td>( z_2 )</td>
<td>( z_{21} z_{22} \ldots z_{2j} )</td>
<td>-0.99</td>
<td>0.72</td>
</tr>
<tr>
<td>( z_3 )</td>
<td>( z_{31} z_{32} \ldots z_{3k} )</td>
<td>-3.76</td>
<td>0.73</td>
</tr>
</tbody>
</table>
Expected F-Measure Training

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>$\gamma(y_i)$</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y_1$</td>
<td>$y_{11} \ y_{12} \ldots \ y_{1i}$</td>
<td>-0.60</td>
<td>0.67</td>
</tr>
<tr>
<td>$y_2$</td>
<td>$y_{21} \ y_{22} \ldots \ y_{2j}$</td>
<td>-1.5</td>
<td>0.81</td>
</tr>
<tr>
<td>$y_3$</td>
<td>$y_{31} \ y_{32} \ldots \ y_{3k}$</td>
<td>-4.96</td>
<td>0.90</td>
</tr>
</tbody>
</table>

$$J(\theta) = -XF1(\theta) = - \sum_{y_i \in \Lambda(x_n)} p(y_i|\theta)F1(\Delta y_i, \Delta^G x_n) = -71.00$$

<table>
<thead>
<tr>
<th>output</th>
<th>action sequence</th>
<th>$\gamma(y_i)$</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z_1$</td>
<td>$z_{11} \ z_{12} \ldots \ z_{1i}$</td>
<td>-0.90</td>
<td>0.71</td>
</tr>
<tr>
<td>$z_2$</td>
<td>$z_{21} \ z_{22} \ldots \ z_{2j}$</td>
<td>-0.99</td>
<td>0.72</td>
</tr>
<tr>
<td>$z_3$</td>
<td>$z_{31} \ z_{32} \ldots \ z_{3k}$</td>
<td>-3.76</td>
<td>0.73</td>
</tr>
</tbody>
</table>

$$J(\theta) = -XF1(\theta) = - \sum_{z_i \in \Lambda(x_n)} p(z_i|\theta)F1(\Delta z_i, \Delta^G x_n) = -71.20$$
Related Work

\[ \text{Perceptron Layer} \]
\[
\arg \max_{y \in \text{GEN}(x)} \sum_{j=1}^{m} v(y_j) \cdot \phi(x, c_j)
\]

\[ \text{Softmax Layer} \]
\[
P(y) \propto \exp(\beta_y^T h_2 + b_y)
\]

\[ \text{Hidden Layers} \]
\[
h_2 = \max\{0, W_2h_1 + b_2\}
\]
\[
h_1 = \max\{0, W_1h_0 + b_1\}
\]

\[ \text{Embedding Layer} \]
\[
h_0 = [X_gE_g] \quad \forall g \in \{\text{word, tag, label}\}
\]

[Weiss et al., 2015]
Related Work

• Watanabe and Sumita, 2015
  – max-margin based objective
  – max-violation updates [Huang et al., 2012]

• Zhou et al., 2015
  – based on Chen and Manning, 2014
  – CRF [Bottou et al., 1997; Le Cun et al., 1998; Lafferty et al., 2001]

• Andor et al., 2016
  – based on Chen and Manning, 2014 and Weiss et al., 2015
  – also CRF
Related Work

• Watanabe and Sumita, 2015
  – max-margin based objective
  – max-violation updates [Huang et al., 2012]

• Zhou et al., 2015
  – based on Chen and Manning, 2014
  – CRF [Bottou et al., 1997; Le Cun et al., 1998; Lafferty et al., 2001]

• Andor et al., 2016
  – based on Chen and Manning, 2014 and Weiss et al., 2015
  – also CRF

• Optimizing task-specific metrics for parsing
  – e.g., Goodman, 1996; Smith and Eisner, 2006; Auli and Lopez, 2011
Eval: F1 over Labeled, Directed CCG Deps

\[
\begin{array}{c}
NP/N \quad N \quad (NP\backslash NP)/(S/NP) \quad NP \quad (S\backslash NP)/NP \\
NP \quad (S/(S\backslash NP)) \quad S/\NP \\
NP\backslash NP \quad NP
\end{array}
\]

\[
\langle \text{the}, NP/N_1, 1, \text{books}, \rangle \\
\langle \text{likes}, (S\backslash NP_1)/NP_2, 1, \text{John} \rangle \\
\langle \text{which}, (NP/N_1)/(S/NP)_2, 2, \text{likes} \rangle \\
\langle \text{which}, (NP/N_1)/(S/NP)_2, 1, \text{books} \rangle \\
\langle \text{likes}, (S\backslash NP_1)/NP_2, 2, \text{books} \rangle
\]
The Greedy Model and Beam Search (Dev)

<table>
<thead>
<tr>
<th>beam</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b = 1$</td>
<td>84.61</td>
</tr>
<tr>
<td>$b = 2$</td>
<td>84.94</td>
</tr>
<tr>
<td>$b = 4$</td>
<td>85.01</td>
</tr>
<tr>
<td>$b = 6$</td>
<td>85.02</td>
</tr>
<tr>
<td>$b = 8$</td>
<td>85.02</td>
</tr>
<tr>
<td>$b = 16$</td>
<td>85.01</td>
</tr>
</tbody>
</table>

$b \in \{6, 8\}$ gives +0.41% F1 over $b = 1$
XF1 Model Dev F1 vs. Training Epochs

![Graph showing F1 on dev set vs. training epochs for RNN-xF1 (b = 8).](image)
## Test Set Parsing Results

<table>
<thead>
<tr>
<th>Model</th>
<th>LP</th>
<th>LR</th>
<th>LF</th>
<th>CAT</th>
<th>Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>C&amp;C (normal)</td>
<td>85.45</td>
<td>83.97</td>
<td>84.70</td>
<td>92.83</td>
<td>97.90</td>
</tr>
<tr>
<td>C&amp;C (hybrid)</td>
<td>86.24</td>
<td>84.17</td>
<td>85.19</td>
<td>93.00</td>
<td>95.25</td>
</tr>
<tr>
<td>Zhang11 ($b = 16$)</td>
<td>87.04</td>
<td>84.14</td>
<td>85.56</td>
<td>92.95</td>
<td>49.54</td>
</tr>
<tr>
<td>Xu14 ($b = 128$)</td>
<td>87.03</td>
<td>85.08</td>
<td>86.04</td>
<td>93.10</td>
<td>12.85</td>
</tr>
<tr>
<td>Am16 ($b = 1$)</td>
<td>-</td>
<td>-</td>
<td>83.27</td>
<td>91.89</td>
<td>350.00</td>
</tr>
<tr>
<td>Am16 ($b = 16$)</td>
<td>-</td>
<td>-</td>
<td>85.57</td>
<td>92.86</td>
<td>10.00</td>
</tr>
<tr>
<td>RNN-greedy ($b = 1$)</td>
<td>88.53</td>
<td>81.65</td>
<td>84.95</td>
<td>93.57</td>
<td>337.45</td>
</tr>
<tr>
<td>RNN-greedy ($b = 6$)</td>
<td>88.54</td>
<td>82.77</td>
<td>85.56</td>
<td>93.68</td>
<td>96.04</td>
</tr>
<tr>
<td>RNN-XF1 ($b = 8$)</td>
<td>88.74</td>
<td>84.22</td>
<td>86.42</td>
<td>93.87</td>
<td>67.65</td>
</tr>
</tbody>
</table>

- Zhang11 = Zhang and Clark, 2011*, Xu14 = [Xu et al., 2014]; AM16 = Ambati et al., 2016 (NN + Struct. Percep [Weiss et al., 2015])
- The XF1 model improves LR by 2.57% and LF by 1.47% over RNN-greedy ($b = 1$)
Model 3

[Xu, EMNLP 2016]
Transition-based Dependency Parsing

Configuration: $c_i \rightarrow c_{i+1}$

Action: $c_i \rightarrow c_{i+1}$

Derivation: $c_0, a_0 \rightarrow c_1, a_1 \rightarrow c_2, a_2$

source: Google SyntaxNet
Models

- Local linear (e.g., SVM)
Models

• Local linear (e.g., SVM) $\Rightarrow$ global linear (e.g., struct. perceptron)
Models

- Local linear (e.g., SVM) ⇒ global linear (e.g., struct. perceptron)

- Local NNs and RNNs

No “global” sensitivity to parser states
Solution: Stack-LSTM [Dyer et al., 2015]
Models

- Local linear (e.g., SVM) $\Rightarrow$ global linear (e.g., struct. perceptron)

- Local NNs and RNNs $\Rightarrow$ global NNs and RNNs (e.g., NNs + CRF [Andor et al., 2016] and XF1)
Models

- Local linear (e.g., SVM) ⇒ global linear (e.g., struct. perceptron)

- Local NNs and RNNs ⇒ global NNs and RNNs (e.g., NNs + CRF [Andor et al., 2016] and XF1)

<table>
<thead>
<tr>
<th>step</th>
<th>stack ((s_n, \ldots, s_1, s_0))</th>
<th>queue ((q_0, q_1 \ldots, q_n))</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
</tbody>
</table>

No “global” sensitivity to parser states
Models

- Local linear (e.g., SVM) $\Rightarrow$ global linear (e.g., struct. perceptron)

- Local NNs and RNNs $\Rightarrow$ global NNs and RNNs (e.g., NNs + CRF [Andor et al., 2016] and XF1)

<table>
<thead>
<tr>
<th>step</th>
<th>stack $(s_n, \ldots, s_1, s_0)$</th>
<th>queue $(q_0, q_1 \ldots, q_n)$</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td>Ms. Haag plays Elianti</td>
</tr>
</tbody>
</table>

No “global” sensitivity to parser states

Solution: Stack-LSTM [Dyer et al., 2015]
Stack-LSTM  [Dyer et al., 2015]

<table>
<thead>
<tr>
<th>step</th>
<th>stack (s_n, ..., s_1, s_0)</th>
<th>queue (q_0, q_1, ..., q_n)</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>Ms. Haag plays Elianti</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>N/N</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>N/N N</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>3</td>
<td>N</td>
<td>plays Elianti</td>
<td>REDUCE</td>
</tr>
<tr>
<td>4</td>
<td>NP</td>
<td>plays Elianti</td>
<td>UNARY</td>
</tr>
<tr>
<td>5</td>
<td>NP  (S[decl]/NP)/NP</td>
<td>Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>6</td>
<td>NP  (S[decl]/NP)/NP N</td>
<td></td>
<td>SHIFT</td>
</tr>
<tr>
<td>7</td>
<td>NP  (S[decl]/NP)/NP NP</td>
<td></td>
<td>UNARY</td>
</tr>
<tr>
<td>8</td>
<td>NP  S[decl]/NP</td>
<td></td>
<td>REDUCE</td>
</tr>
<tr>
<td>9</td>
<td>S[decl]</td>
<td></td>
<td>REDUCE</td>
</tr>
</tbody>
</table>
Stack-LSTM [Dyer et al., 2015]

<table>
<thead>
<tr>
<th>step</th>
<th>stack</th>
<th>queue</th>
<th>action</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$(s_n, \ldots, s_1, s_0)$</td>
<td>$(q_0, q_1 \ldots, q_n)$</td>
<td>Ms. Haag plays Elianti</td>
</tr>
<tr>
<td>1</td>
<td>$N/N$</td>
<td>Haag plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>2</td>
<td>$N/N N$</td>
<td>plays Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>3</td>
<td>$N$</td>
<td>plays Elianti</td>
<td>REDUCE</td>
</tr>
<tr>
<td>4</td>
<td>$NP$</td>
<td>plays Elianti</td>
<td>UNARY</td>
</tr>
<tr>
<td>5</td>
<td>$NP (S[dcl]\backslash NP)/NP$</td>
<td>Elianti</td>
<td>SHIFT</td>
</tr>
<tr>
<td>6</td>
<td>$NP (S[dcl]\backslash NP)/NP N$</td>
<td></td>
<td>SHIFT</td>
</tr>
<tr>
<td>7</td>
<td>$NP (S[dcl]\backslash NP)/NP NP$</td>
<td></td>
<td>UNARY</td>
</tr>
<tr>
<td>8</td>
<td>$NP S[dcl]\backslash NP$</td>
<td></td>
<td>REDUCE</td>
</tr>
<tr>
<td>9</td>
<td>$S[dcl]$</td>
<td></td>
<td>REDUCE</td>
</tr>
</tbody>
</table>

LSTM-stack, LSTM-queue, LSTM-action
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
Stack-LSTM [Dyer et al., 2015]

It showed promise
the books which John likes
the books which John likes

NP/N
LSTM Shift-Reduce CCG Parsing

the books which John likes

NP/N N

W→□→□
C→□→□
P→□→□
A→□→□
the books which John likes
LSTM Shift-Reduce CCG Parsing

the books which John likes

NP/N N (NP\NP)/(S/NP)

W → □ → □ → □ → □
C → □ → □ → □ → □
P → □ → □ → □ → □
A → □ → □ → □

□ → □ → □ → □
□ → □ → □ → □
□ → □ → □ → □
□ → □ → □ → □
LSTM Shift-Reduce CCG Parsing

the books which John likes

NP/N  N  (NP\NP)/(S/NP)  NP
LSTM Shift-Reduce CCG Parsing

\[
\delta_t = \left[ h_W t; h_C t; h_P t; h_A t \right] b_t = f(B[\delta_t; Qj] + r)
\]

\[
a_t = f(Ab_t + s)
\]
LSTM Shift-Reduce CCG Parsing

\[ \delta_t = [h_t^W; h_t^C; h_t^P; h_t^A] \]
LSTM Shift-Reduce CCG Parsing

\[ \delta_t = [h^W_t; h^C_t; h^P_t; h^A_t] \]

\[ b_t = f(B[\delta_t; Q_j] + r) \]
LSTM Shift-Reduce CCG Parsing

\[ \delta_t = [h_t^W; h_t^C; h_t^P; h_t^A] \]

\[ b_t = f(B[\delta_t; Q_j] + r) \]

\[ a_t = f(Ab_t + s) \]
Two Simple Motivations: I

He learned some French and German

this parser

He learned some French and German

Google SyntaxNet and Stanford
Two Simple Motivations: II

input : $w_0 \ldots w_{n-1}$

axiom : $0 : (0, \epsilon, \beta, \phi)$

goal : $2n - 1 + \mu : (n, \delta, \epsilon, \Delta)$

\[
\begin{align*}
\omega : (j, \delta, x_{w_j} | \beta, \Delta) \\
\omega + 1 : (j + 1, \delta | x_{w_j}, \beta, \Delta)
\end{align*}
\]

(SHIFT; $0 \leq j < n$)

\[
\begin{align*}
\omega : (j, \delta | s_1 s_0, \beta, \Delta) \\
\omega + 1 : (j, \delta | x, \beta, \Delta \cup \langle x \rangle)
\end{align*}
\]

(REDUCE; $s_1 s_0 \rightarrow x$)

\[
\begin{align*}
\omega : (j, \delta | s_0, \beta, \Delta) \\
\omega + 1 : (j, \delta | x, \beta, \Delta)
\end{align*}
\]

(UNARY; $s_0 \rightarrow x$)
Results: Locally Normalized Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSTM-W</td>
<td>83.05</td>
</tr>
<tr>
<td>LSTM-W+C</td>
<td>86.20</td>
</tr>
<tr>
<td>LSTM-W+C+A</td>
<td>86.25</td>
</tr>
<tr>
<td>LSTM-W+C+A+P</td>
<td>86.56</td>
</tr>
</tbody>
</table>
Results: Locally Normalized Models

![Graph showing F1 (labeled) on dev set against training epochs for LSTM-w, LSTM-wc, LSTM-wca, and LSTM-wcap models. The y-axis represents F1 scores ranging from 78 to 87, and the x-axis represents training epochs from 0 to 30. The LSTM-w model shows a steady increase in F1 score, while LSTM-wc, LSTM-wca, and LSTM-wcap exhibit fluctuations before converging.]
Results: Locally Normalized Models

<table>
<thead>
<tr>
<th>Model</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-XENT (B = 1)</td>
<td>84.95</td>
</tr>
<tr>
<td>RNN-XF1 (B = 8)</td>
<td>86.42</td>
</tr>
<tr>
<td>LSTM-XENT (B = 1)</td>
<td>86.56</td>
</tr>
</tbody>
</table>
The Label Bias Problem

[Bottou et al., 1997; LeCun et al., 1998; Lafferty et al., 2001]

\[
p(y_t | \langle s, q \rangle_y^{t-1}; \theta) = \frac{\exp\{\gamma(y_t, \langle s, q \rangle_y^{t-1}; \theta)\}}{Z_L(\langle s, q \rangle_y^{t-1})}
\]

\[
Z_L(\langle \alpha, \beta \rangle_y^{t-1}) = \sum_{y_t' \in \mathcal{T}(\langle \alpha, \beta \rangle_y^{t-1})} \exp\{\gamma(y_t', \langle \alpha, \beta \rangle_y^{t-1}; \theta)\}
\]

Andor et al., (2016) showed that \( \mathcal{P}_L \subset \mathcal{P}_G \)
The Label Bias Problem

[Bottou et al., 1997; LeCun et al., 1998; Lafferty et al., 2001]

\[ p(y_t|\langle s, q\rangle_{y}^{t-1}; \theta) = \frac{\exp\{\gamma(y_t, \langle s, q\rangle_{y}^{t-1}; \theta)\}}{Z_L(\langle s, q\rangle_{y}^{t-1})} \]

\[ Z_L(\langle \alpha, \beta\rangle_{y}^{t-1}) = \sum_{y_t' \in T(\langle \alpha, \beta\rangle_{y}^{t-1})} \exp\{\gamma(y_t', \langle \alpha, \beta\rangle_{y}^{t-1}; \theta)\} \]

Andor et al., (2016) showed that \( P_L \subset P_G \) and label bias is irrespective of the scoring function \( \gamma \)
XF1 Training

\[ h_1 \rightarrow h_3 \rightarrow h_7 \rightarrow h_{15} \rightarrow h_{25} \]

\[ h_2 \rightarrow h_6 \rightarrow h_{12} \rightarrow h_{10} \rightarrow y_1 \]

\[ h_4 \rightarrow h_8 \rightarrow h_{14} \rightarrow h_{22} \rightarrow y_2 \]

\[ h_{10} \rightarrow h_{20} \rightarrow y_3 \]

\[ h_{12} \rightarrow h_{22} \rightarrow y_4 \]
XF1 Training

W → □ → □ → □ → □ → □
C → □ → □ → □ → □ → □
P → □ → □ → □ → □ → □
A → □ → □ → □ → □ → □ → □ → □

□ → □ → □ → □ → □ → □ → □ → □ → □ → □ → □ → □ → □ → □
XF1 Training
Results: XF1 Models

F1 (labeled) on dev set vs. Training epochs for LSTM-XF1 (beam = 8)
Results: XF1 Models

- LSTM-XENT: 86.83
- LSTM-XF1 (B = 1): 87.62
- LSTM-XF1 (B = 8): 87.76
Impl.: Tree-Structured Stack + Dynamically Structured Graph
\[ \delta^a_{s_r} = \text{BPTS}(s_r.A) \]
\[ = \sum_{m \in s_r.A.\text{keys}} \sum_{i \in s_r.A[m]} \delta_m \delta_{im} \]
\[ = \sum_{m \in s_r.A.\text{keys}} \sum_{i \in s_r.A[m]} \delta_m p(y_i|\theta)(XF1(\theta) - F1(\Delta y_i, \Delta^G_{x_n})) \frac{1}{Z_m} \]

**XF1 gradient per action**
Impl.: Tree-Structured Stack + Dynamically Structured Graph
Conclusions

• Global normal-form
Conclusions

- Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)

- Local LSTM with global sensitivity $\Rightarrow$ global LSTM (optimized for the evaluation metric)

- Beam search $\Rightarrow$ struct. perceptron, RNN, and LSTM $\Rightarrow$ global structured learning
Conclusions

- Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN
Conclusions

• Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

• Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)
Conclusions

• Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

• Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)

• Local LSTM with global sensitivity
Conclusions

- Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)

- Local LSTM with global sensitivity $\Rightarrow$ global LSTM (optimized for the evaluation metric)
Conclusions

- Global normal-form ⇒ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN ⇒ global RNN (optimized for the evaluation metric)

- Local LSTM with global sensitivity ⇒ global LSTM (optimized for the evaluation metric)

- Beam search
Conclusions

- Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)

- Local LSTM with global sensitivity $\Rightarrow$ global LSTM (optimized for the evaluation metric)

- Beam search $\Rightarrow$ struct. perceptron, RNN, and LSTM
Conclusions

- Global normal-form $\Rightarrow$ global dependency model with a hidden variable (with the struct. perceptron)

- Local RNN $\Rightarrow$ global RNN (optimized for the evaluation metric)

- Local LSTM with global sensitivity $\Rightarrow$ global LSTM (optimized for the evaluation metric)

- Beam search $\Rightarrow$ struct. perceptron, RNN, and LSTM $\Rightarrow$ global structured learning