Recursive Subtree Composition in LSTM-Based Dependency Parsing

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre

4 June 2019
Overview

1. Tree vs. sequential LSTMs for parsing
2. BiLSTM parsing
3. Results
4. Conclusion
Outline for section 1

1. Tree vs. sequential LSTMs for parsing
2. BiLSTM parsing
3. Results
4. Conclusion
The largest city in Minnesota

Recursive

The largest city in Minnesota

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre
Recursive Subtree Composition in Parsing 4/22
Recursive vs recurrent NNs

Recursive

The largest city in Minnesota

Recurrent

The largest city in Minnesota

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre

Recursive Subtree Composition in Parsing
the largest city
Recursive NN for Transition-Based Parsing

the largest city

left-arc

nmod
Recursive NN for Transition-Based Parsing

the largest city

left-arc

det

nmod

the

largest

city
Recursive composition function in the stack-LSTM parser (Dyer et al., 2015):
Recursive composition function in the stack-LSTM parser (Dyer et al., 2015):

\[ c(h, d, r) = \tanh(W[h; d; r] + b) \]
Recursive composition function in the stack-LSTM parser (Dyer et al., 2015):

\[ c(h, d, r) = \tanh(W[h; d; r] + b) \]
\[ h_i = c(h_{i-1}, d, r) \]
Recursive composition function in the stack-LSTM parser (Dyer et al., 2015):

\[ c(h, d, r) = \tanh(W[h; d; r] + b) \]

\[ city_1 = c(city_0, largest, left \text{–} nmod) \]
Recursive composition function in the stack-LSTM parser (Dyer et al., 2015):

\[ c(h, d, r) = \tanh(W[h; d; r] + b) \]

\[ city_1 = c(city_0, \text{largest}, \text{left} \, - \, n\text{mod}) \]

\[ city_2 = c(city_1, \text{the}, \text{left} \, - \, d\text{et}) \]
Recursive vs recurrent NNs

S-LSTM without composition
89.6 83.6

S-LSTM with composition
90.9 85.7

BiLSTM
91.2 85.0
Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th></th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>91.2</td>
<td>85.0</td>
</tr>
</tbody>
</table>
## Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th></th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
</tbody>
</table>

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre
<table>
<thead>
<tr>
<th>Model</th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
</tbody>
</table>
## Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th>Model</th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>91.2</td>
<td>85.0</td>
</tr>
</tbody>
</table>
# Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th>Model</th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>91.2</td>
<td>85.0</td>
</tr>
</tbody>
</table>

**Goals**

- BiLSTM + composition?
- Examine composition in simple architecture
- Typologically varied languages
Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th>Model</th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>91.2</td>
<td>85.0</td>
</tr>
</tbody>
</table>

Goals

- BiLSTM + composition?
### Goals
- BiLSTM + composition?
- Examine composition in simple architecture
## Recursive vs recurrent NNs

<table>
<thead>
<tr>
<th>Model</th>
<th>English PTB</th>
<th>Chinese CTB</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-LSTM without composition</td>
<td>89.6</td>
<td>83.6</td>
</tr>
<tr>
<td>S-LSTM with composition</td>
<td>90.9</td>
<td>85.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>91.2</td>
<td>85.0</td>
</tr>
</tbody>
</table>

**Goals**

- BiLSTM + composition?
- Examine composition in simple architecture
- Typologically varied languages
Outline for section 2

1. Tree vs. sequential LSTMs for parsing
2. BiLSTM parsing
3. Results
4. Conclusion
Transition-Based Parsing using BiLSTMs

Configuration:

STACK

the brown fox

BUFFER

jumped root

Scoring:

(score(LEFT-ARC), score(RIGHT-ARC), score(SHIFT), score(SWAP))

Kiperwasser and Goldberg (2016); de Lhoneux et al. (2017)
Xthe
Transition-Based Parsing using BiLSTMs

\[
\begin{array}{c}
\text{Xthe} \\
\text{e(the)}
\end{array}
\]
Transition-Based Parsing using BiLSTMs

Recursive Subtree Composition in Parsing

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre
Transition-Based Parsing using BiLSTMs
Transition-Based Parsing using BiLSTMs

X_the X_brown X_fox X_jumped X_root
Transition-Based Parsing using BiLSTMs
Transition-Based Parsing using BiLSTMs
Transition-Based Parsing using BiLSTMs
(score(LEFT–ARC), score(RIGHT–ARC), score(SHIFT), score(SWAP))
Recursive Composition in the BiLSTM parser

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre
Recursive Subtree Composition in Parsing
Recursive Composition in the BiLSTM parser

Recursive Subtree Composition in Parsing

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre
Recursive Composition in the BiLSTM parser
Recursive Composition in the BiLSTM parser
Recursive Composition in the BiLSTM parser

$$C_{fox} = \tanh(W[C_{fox}, C_{brown}, left-nmod] + b)$$
Recursive Composition in the BiLSTM parser

The diagram illustrates the process of recursive composition in the BiLSTM parser. The network processes a sentence (e.g., "the brown fox jumped root") and uses LSTM layers to encode the input sequence. Each word or phrase is fed into the LSTM layers, which then combine their representations through concatenation. The output of each LSTM layer is fed into the next, allowing for hierarchical processing of the sentence structure.

The diagram shows a sentence "the brown fox jumped root" with the corresponding LSTM layers and concatenation operations.
Recursive Composition in the BiLSTM parser

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre

Recursive Subtree Composition in Parsing 11/22
Recursive Composition in the BiLSTM parser

\[ c_{head} = \tanh(W[h; d; r] + b) \]
Recursive Composition in the BiLSTM parser

\[ c_{\text{head}} = \tanh(W[h; d; r] + b) + rc \]
Recursive Composition in the BiLSTM parser

\[ c_{head} = \tanh(W[h; d; r] + b) + rc \]

\[ c_{head} = \text{LSTM}([h; d; r]) \]
Recursive Composition in the BiLSTM parser

\[ c_{\text{head}} = \tanh(W[h; d; r] + b) + rc \]

\[ c_{\text{head}} = \text{LSTM}([h; d; r]) + lc \]
Outline for section 3

1. Tree vs. sequential LSTMs for parsing
2. BiLSTM parsing
3. Results
4. Conclusion
Results: BiLSTM + composition
Results: BiLSTM + composition

![Bar Chart with Error LAS for Different Languages]

- **Languages**: cs, en, eu, fi, fr, grc, he, ja, zh, av.
- **Comparison**: bi vs. bi+rc
- **Error LAS**: Y-axis represents the error LAS.

---

Miryam de Lhoneux, Miguel Ballesteros and Joakim Nivre  
Recursive Subtree Composition in Parsing  
14/22
Results: BiLSTM + composition

![Bar chart showing error rates for various languages using BiLSTM + composition.](chart.png)
LSTM Feature Extractors

![Diagram of LSTM feature extractors with inputs and outputs](attachment:diagram.png)
LSTM Feature Extractors
LSTM Feature Extractors

Vthe \rightarrow LSTM^b \rightarrow X_{\text{the}}

Vbrown \rightarrow LSTM^b \rightarrow X_{\text{brown}}

Vfox \rightarrow LSTM^b \rightarrow X_{\text{fox}}

Vjumped \rightarrow LSTM^b \rightarrow X_{\text{jumped}}

Vroot \rightarrow LSTM^b \rightarrow X_{\text{root}}

bw
LSTM Feature Extractors

\[ \text{Vthe} \quad \text{Vbrown} \quad \text{Vfox} \quad \text{Vjumped} \quad \text{Vroot} \]

\[ \xrightarrow{\text{LSTM}} \quad \xrightarrow{\text{LSTM}} \quad \xrightarrow{\text{LSTM}} \quad \xrightarrow{\text{LSTM}} \quad \xrightarrow{\text{LSTM}} \]

\[ X_{\text{the}} \quad X_{\text{brown}} \quad X_{\text{fox}} \quad X_{\text{jumped}} \quad X_{\text{root}} \]

\[ f_{\text{w}} \]
Results: BiLSTM ablations

![Graph showing error LAS for different languages]
Results: BiLSTM ablations
Results: BiLSTM ablations
Results: BiLSTM ablations + composition

![Bar chart showing error LAS for different compositions.](chart.png)
Results: BiLSTM ablations + composition

![Bar chart showing error rates for BiLSTM ablations + composition]

- bi
- bw
- fw
- bi+rc
- bw+rc
- fw+rc

Error LAS

av.
Results: BiLSTM ablations + composition

![Bar chart showing results of BiLSTM ablations + composition.](image)
Word representation

\[Xthe\]

\[
e(\text{the}) \quad \text{pe(\text{the})}
\]

\[
\text{concat}
\]

\[
\text{Cf} \quad \text{Cf} \quad \text{Cf}
\]

\[
\text{Cb} \quad \text{Cb} \quad \text{Cb}
\]

\[
t \quad \text{h} \quad \text{e}
\]
Word representation

Xthe

+pos

e(the)  pe(the)

concat

Cb  Cb  Cb

t  h  e
Word representation

Xthe

e(the)  pe(the)

concat

+Cf
+Cf
+Cf

+char

t  h  e

+Cb
+Cb
+Cb
## Composition gap recovery

<table>
<thead>
<tr>
<th></th>
<th>[bw+lc]-bw</th>
<th>[fw+lc]-fw</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pos+char+</strong></td>
<td>1.4</td>
<td>0.6</td>
</tr>
<tr>
<td><strong>pos+char-</strong></td>
<td>1.3</td>
<td>0.6</td>
</tr>
<tr>
<td><strong>pos-char+</strong></td>
<td>1.6</td>
<td>0.7</td>
</tr>
<tr>
<td><strong>pos-char-</strong></td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

av.
## Composition gap recovery

<table>
<thead>
<tr>
<th></th>
<th>[bw+lc]-bw</th>
<th>bi-bw</th>
<th>%rec.</th>
<th>[fw+lc]-fw</th>
<th>bi-fw</th>
<th>%rec.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pos+char⁺</strong></td>
<td>1.4</td>
<td>1.6</td>
<td><strong>87.5</strong></td>
<td>0.6</td>
<td>6.3</td>
<td><strong>9.5</strong></td>
</tr>
<tr>
<td><strong>pos+char⁻</strong></td>
<td>1.3</td>
<td>1.8</td>
<td><strong>72.2</strong></td>
<td>0.6</td>
<td>6.6</td>
<td><strong>9.1</strong></td>
</tr>
<tr>
<td><strong>pos-char⁺</strong></td>
<td>1.6</td>
<td>1.9</td>
<td><strong>84.2</strong></td>
<td>0.7</td>
<td>7.3</td>
<td><strong>9.6</strong></td>
</tr>
<tr>
<td><strong>pos-char⁻</strong></td>
<td>2</td>
<td>3.1</td>
<td><strong>64.5</strong></td>
<td>1</td>
<td>8.7</td>
<td><strong>11.5</strong></td>
</tr>
</tbody>
</table>

**av.**
Outline for section 4

1. Tree vs. sequential LSTMs for parsing
2. BiLSTM parsing
3. Results
4. Conclusion
Subtree composition does not reliably help a BiLSTM transition-based parser.
**Conclusion**

- Subtree composition does not reliably help a BiLSTM transition-based parser.
- The backward part of the BiLSTM is crucial, especially for right-headed languages.
Conclusion

- Subtree composition does not reliably help a BiLSTM transition-based parser
- The backward part of the BiLSTM is crucial, especially for right-headed languages
- The forward part of the BiLSTM is less crucial
Conclusion

- Subtree composition does not reliably help a BiLSTM transition-based parser
- The backward part of the BiLSTM is crucial, especially for right-headed languages
- The forward part of the BiLSTM is less crucial
- A backward LSTM + subtree composition performs close to a BiLSTM
Conclusion

- Subtree composition does not reliably help a BiLSTM transition-based parser.
- The backward part of the BiLSTM is crucial, especially for right-headed languages.
- The forward part of the BiLSTM is less crucial.
- A backward LSTM + subtree composition performs close to a BiLSTM.
- POS information and subtree composition are two partially redundant ways of constructing contextual information.
