A Transition-Based Directed Acyclic Graph Parser for Universal Conceptual Cognitive Annotation

Daniel Hershcovich, Omri Abend and Ari Rappoport

Technion
June 29, 2017
Linguistic Annotation Schemes

- Dependency grammar (Tesnière, 1959; Nivre, 2005):
  Purely syntactic analysis.

- Semantic representation (Abend and Rappoport, 2017):

- Semantic dependencies (Oepen et al., 2015):
  Coupled with syntactic representation.

- AMR (Banarescu et al., 2013), UCCA (Abend and Rappoport, 2013):
  \[ \ldots \text{showered} = \ldots \text{took a shower} \]
  \[ \ldots \text{’s war against crime} = \ldots \text{fights crime} \]
Dependency Parsing

- Graph representation of syntactic structure.
- Bilexical tree: edges are only between tokens.
- Fast and accurate parsers (e.g. \textit{transition-based}).

Non-projectivity (discontinuity) is a challenge (Nivre, 2009).

A hearing is scheduled on the issue today.
Dependency Parsing

- Graph representation of syntactic structure.
- Bilexical tree: edges are only between tokens.
- Fast and accurate parsers (e.g. transition-based).

Non-projectivity (discontinuity) is a challenge (Nivre, 2009).
Semantic Dependency Parsing

- Representation of predicate-argument relationships.
- Bilexical graph: allows reentrancy (and discontinuity).
- Various formalisms.

After graduation, Joe moved to Paris

DELPH-IN MRS-derived bi-lexical dependencies (DM).
Semantic Dependency Parsing

- Representation of predicate-argument relationships.
- Bilexical graph: allows reentrancy (and discontinuity).
- Various formalisms.

**DELPH-IN MRS-derived bi-lexical dependencies (DM).**

**Prague Dependency Treebank tectogrammatical layer (PSD).**
After graduation, Joe moved to Paris.
The UCCA Semantic Representation Scheme

Cross-linguistically applicable (Abend and Rappoport, 2013).
Stable in translation (Sulem et al., 2015).

IBM happened to choose a company with a crucial vulnerability, despite vetting.
UCCAApp


William Bradley Pitt was born in Shawnee, Oklahoma, to William Alvin Pitt, who ran a trucking company, and Jane Etta (née Hillhouse), a school counsellor. The family soon moved to Springfield, Missouri, where he lived together with his younger siblings, Douglas (born 1966) and Julie Neal (born 1969). Born into a conservative household, he was raised as Southern Baptist, but has since stated that he does not have a great relationship with religion and that he oscillates between agnosticism and atheism. Pitt has described Springfield as "Mark Twain country", Jesse James country, having grown up with a lot of hills, a lot of lakes."
UCCA facilitates semantics-based human evaluation of machine translation (Birch et al., 2016). http://ucca.cs.huji.ac.il/mteval

For mildly obese diabetics, weight loss surgery may be helpful.
Graph Structure

UCCA forms a directed acyclic graph (DAG). Tokens are terminals. Structural properties:

1. Non-terminal nodes

You want to take a long bath
Graph Structure

UCCA forms a directed acyclic graph (DAG). Tokens are terminals.

- **Structural properties:**
  1. **Non-terminal nodes**
  2. **Reentrancy**

You want to take a long bath
Graph Structure

UCCA forms a directed acyclic graph (DAG). Tokens are terminals. Structural properties:

1. Non-terminal nodes
2. Reentrancy
3. Discontinuity

You want to take a long bath
Transition-Based Parsing

- Parse text $w_1 \ldots w_n$ to graph $G = (V, E, \ell)$ incrementally.
- Classifier determines transition to apply at each step.
- Trained by an oracle based on gold-standard graph.
Transition-Based Parsing

- Parse text $w_1 \ldots w_n$ to graph $G = (V, E, \ell)$ incrementally.
- Classifier determines transition to apply at each step.
- Trained by an oracle based on gold-standard graph.

Initial state:

stack

buffer

\[
\begin{array}{cccccc}
|\bullet| & \text{You} & \text{want} & \text{to} & \text{take} & \text{a} \\
\hline
& \text{a} & \text{long} & \text{bath} & \\
\end{array}
\]
Transition-Based Parsing

- Parse text $w_1 \ldots w_n$ to graph $G = (V, E, \ell)$ incrementally.
- Classifier determines transition to apply at each step.
- Trained by an oracle based on gold-standard graph.

Initial state:

```
stack                                buffer
● You want to take a long bath
```

TUPA transitions:

```
{Shift, Reduce, Node_X, Left-Edge_X, Right-Edge_X, 
  Left-Remote_X, Right-Remote_X, Swap, Finish}
```

Support non-terminal nodes, reentrancy and discontinuity.
Transition-Based UCCA Parsing

⇒ **SHIFT**

stack

buffer

<table>
<thead>
<tr>
<th>dot</th>
<th>You</th>
</tr>
</thead>
</table>

| want | to | take | a | long | bath |

graph

| dot |       |
Transition-Based UCCA Parsing

⇒ \text{RIGHT-EDGE}_A

stack

\begin{array}{|c|}
\hline
\bullet & \text{You} \\
\hline
\end{array}

buffer

\begin{array}{|c|c|c|c|}
\hline
\text{want} & \text{to} & \text{take} & \text{a} \\
\hline
\text{long} & \text{bath} \\
\hline
\end{array}

graph

You

A
Transition-Based UCCA Parsing

⇒ **SHIFT**

stack

<table>
<thead>
<tr>
<th></th>
<th>You</th>
<th>want</th>
</tr>
</thead>
</table>

buffer

| to | take | a | long | bath |

graph

<table>
<thead>
<tr>
<th></th>
<th>You</th>
</tr>
</thead>
</table>

A
Transition-Based UCCA Parsing

⇒ SWAP

stack

buffer

You want to take a long bath

graph

You

A
Transition-Based UCCA Parsing

\[ \Rightarrow \text{RIGHT-EDGE}_P \]

stack

| ● | want |

buffer

| You | to | take | a | long | bath |

graph

You want

\[ \text{A} \quad \text{P} \]
Transition-Based UCCA Parsing

\[ \Rightarrow \text{REDUCE} \]

stack

buffer

to take a long bath

graph

You want

A

P
Transition-Based UCCA Parsing

⇒ $\text{SHIFT}$

stack

buffer

You

to
take

to
long
bath

to
take
long
bath

You

A

P

You

want

graph
Transition-Based UCCA Parsing

⇒ \textbf{SHIFT}

stack

| ● | You | to |

buffer

| take | a | long | bath |

graph

\[
\begin{array}{c}
\text{You} \\
\downarrow \\
\text{want}
\end{array}
\]

\[
\begin{array}{c}
P \\
A
\end{array}
\]
Transition-Based UCCA Parsing

\[ \Rightarrow \text{NODE}_F \]

stack

| ● | You | to |

buffer

| ● | take | a | long | bath |

graph

You \[ \rightarrow A \rightarrow P \rightarrow \text{want} \]

to \[ \rightarrow F \rightarrow \text{to} \]
Transition-Based UCCA Parsing

⇒ REDUCE

stack

| ● | You |

buffer

| ● | take | a | long | bath |

graph

You \(\rightarrow\) P \(\rightarrow\) want

A

F

to
Transition-Based UCCA Parsing

$\Rightarrow$ **SHIFT**

**stack**

- You
- buffer
  - take
  - a
  - long
  - bath

**graph**

- You
- want
  - to
Transition-Based UCCA Parsing

⇒ \textbf{SHIFT}

stack

| ● | You | ● | take |

buffer

| a | long | bath |

graph

You \hspace{1cm} want \hspace{1cm} \text{to}
Transition-Based UCCA Parsing

⇒ NODEC

stack

buffer

You  take

You  want

a  long  bath

take

to

to

graph

A  P

F

C
Transition-Based UCCA Parsing

⇒ REDUCE

stack

buffer

You  a  long  bath

You  want  to  take

A  P  F  C

d graph
Transition-Based UCCA Parsing

⇒ \textbf{SHIFT}

stack

\[
\begin{array}{ccc}
\bullet & \text{You} & \bullet & \bullet
\end{array}
\]

buffer

\[
\begin{array}{ccc}
a & \text{long} & \text{bath}
\end{array}
\]

graph

\[
\begin{tikzpicture}
\node (A) at (0,0) {You};
\node (P) at (1,1) {want};
\node (F) at (2,2) {to};
\node (C) at (3,3) {take};
\draw [->] (A) -- (P) node [midway, above] {A};
\draw [->] (P) -- (F) node [midway, above] {P};
\draw [->] (F) -- (C) node [midway, above] {F};
\end{tikzpicture}
\]
Transition-Based UCCA Parsing

⇒ \texttt{RIGHT-EDGE}_P

stack

\begin{tabular}{|c|c|c|}
\hline
\textbullet & You & \textbullet & \textbullet \\
\hline
\end{tabular}

buffer

\begin{tabular}{|c|c|c|}
\hline
a & long & bath \\
\hline
\end{tabular}

You want to take a long bath.

graph

You \quad \text{want} \quad \to \quad \text{take}

A \quad P \quad F \quad P

C
Transition-Based UCCA Parsing

⇒ SHIFT

stack

buffer

You
a

graph

You
want

take

long
bath
**Transition-Based UCCA Parsing**

\[ \Rightarrow \text{RIGHT-EDGE}_{F} \]

<table>
<thead>
<tr>
<th>stack</th>
<th>buffer</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="stack" /></td>
<td><img src="image" alt="buffer" /></td>
</tr>
</tbody>
</table>

**Graph**

- You \( \xleftarrow{A} P \) want
- You \( \xrightarrow{P} \) to
- You \( \xrightarrow{C} F \) take
- You \( \xrightarrow{F} \) a
- You \( \xleftarrow{P} \) F
- You \( \xrightarrow{C} \) a

You want to take a long bath.
Transition-Based UCCA Parsing

⇒ REDUCE

stack

You

buffer

long bath

graph

A

P

You

want

to

take

a

F

P

C

F

P

C

F

A

P

You

want
Transition-Based UCCA Parsing

⇒ **SHIFT**

stack

<table>
<thead>
<tr>
<th></th>
<th>You</th>
<th></th>
<th>long</th>
</tr>
</thead>
</table>

buffer

bath

graph

You \(\rightarrow\) want \(\rightarrow\) take \(\rightarrow\) a

You \(\rightarrow\) F \(\rightarrow\) to \(\rightarrow\) P

You \(\rightarrow\) P \(\rightarrow\) C \(\rightarrow\) F
Transition-Based UCCA Parsing

⇒ SWAP

stack

buffer

- You
- long

- bath

graph

- You
- want
- to
- a
- take

- P
- F
- C
- A
Transition-Based UCCA Parsing

⇒ \textbf{RIGHT-EDGE}_D

stack

buffer

\begin{center}
\begin{tabular}{|c|c|c|}
\hline
You & long & bath \\
\hline
\end{tabular}
\end{center}

graph

You \quad \text{want} \quad \text{to} \quad \text{take} \quad \text{a} \quad \text{long}
Transition-Based UCCA Parsing

⇒ REDUCE

stack

buffer

[ ] You [ ] bath

graph

A

P

You

want

F

P

to

take

a

long

D
Transition-Based UCCA Parsing

⇒ SWAP

stack

buffer

You  bath

graph

You  want

You  take  a  long

to

A  P

F  P  D

F  C  F
Transition-Based UCCA Parsing

⇒ RIGHT-EDGE$_A$

stack

buffer

You    bath

You want to take a long bath

diagram
Transition-Based UCCA Parsing

⇒ REDUCE

stack

buffer

You

bath

You want to take a long bath
Transition-Based UCCA Parsing

⇒ REDUCE

stack

buffer

You

bath

You

want

A

P

A

to

F

P

take

a

long

D
Transition-Based UCCA Parsing

⇒ **SHIFT**

stack

| You |

buffer

| bath |

graph

```
graph
```

```
You
```

```
want
```

```
A
```

```
P
```

```
A
```

```
You
```

```
want
```

```
F
```

```
P
```

```
to
```

```
C
```

```
F
```

```
take
```

```
a
```

```
long
```

```
P
```

```
D
```

Transition-Based UCCA Parsing

⇒ SHIFT

You

buffer

bath

take a

to

A

A

P

P

F

F

C

D

long

a

to

want

You

graph
Transition-Based UCCA Parsing

\[ \Rightarrow \text{LEFT-REMOTE}_A \]

stack

buffer

You

bath

digraph

You

want

to

take

a

long
Transition-Based UCCA Parsing

⇒ \textbf{SHIFT}

stack

| You | bath |

buffer

graph

You \quad want \quad to \quad take \quad a \quad long
Transition-Based UCCA Parsing

⇒ **RIGHT-EDGE}_C

stack

You  bath

buffer

graph

You  want  a  long  bath

You  want  to  a  long  bath

You  want  to  a  long  bath
Transition-Based UCCA Parsing

⇒ FINISH

stack

You  bath

buffer

graph

You  want  A
    ↓     ↓     ↓
    P     A     A

You  take  to  A
    ↓     ↓     ↓
    F     P     D
    C     F     C

take  a  long  bath
Word Embeddings

Represent discrete features by dense vectors (Goldberg, 2016).

\[ \mathbf{x} = (0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0, 1, 0, ..., 0, 0, 0, ..., 0) \]

\[ \mathbf{x} = (0.26, 0.25, -0.39, -0.07, 0.13, -0.17) \quad (-0.43, -0.37, -0.12, 0.13, -0.11, 0.34) \quad (-0.04, 0.50, 0.04, 0.44) \]

Word Embeddings

POS Embeddings

- Chair: (-0.37, -0.23, 0.33, 0.38, -0.02, -0.37)
  (-0.21, -0.11, -0.10, 0.07, 0.37, 0.15)
  (0.26, 0.25, -0.39, -0.07, 0.13, -0.17)
  ...
  ...
- The: (-0.43, -0.37, -0.12, 0.13, -0.11, 0.34)
  ...
- Mouth: (-0.32, 0.43, -0.14, 0.50, -0.13, -0.42)
  ...
- Gone: (0.06, -0.21, -0.38, -0.28, -0.16, -0.44)
  ...
- Noun: (0.16, 0.03, -0.17, -0.13)
  (0.41, 0.08, 0.44, 0.02)
  ...
- Verb: ...
- Det: (-0.04, 0.50, 0.04, 0.44)
- Adj: (-0.01, -0.35, -0.27, 0.20)
- Prep: (-0.26, 0.28, -0.34, -0.02)
- Adv: (0.02, -0.17, 0.46, -0.08)
  ...
  ...
Feed-Forward Neural Network (MLP)

Learns representation and classification by optimizing weights.
Recurrent Neural Network (RNN)

Applied to sequences, updates state given input and previous state.
Long Short-Term Memory (LSTM)

Memory cell to avoid vanishing gradients in RNNs.
TUPA Model

Greedy parsing, experimenting with three classifiers:

**Sparse**  Perceptron with sparse features.
**MLP**  Embeddings + feedforward NN classifier.
**BiLSTM**  Embeddings + deep bidirectional LSTM + MLP.

Features: words, POS, syntactic dependencies, existing edge labels from the stack and buffer + parents, children, grandchildren; ordinal features (height, number of parents and children)
BiLSTM

You want to take a long bath.
You want to take a long bath.
BiLSTM

You want to take a long bath
You want to take a long bath.
Experimental Setup

- UCCA Wikipedia corpus \((4268 + 454 + 503\) sentences).
Evaluation

Comparing graphs over the same sequence of tokens,
• Match edges by their terminal yield and label.
• Calculate labeled precision, recall and F1 scores.
• Separate primary and remote edges.

Primary: \[ \frac{6}{9} = 67\% \quad \frac{6}{10} = 60\% \quad 64\% \]

Remote: \[ \frac{1}{2} = 50\% \quad \frac{1}{1} = 100\% \quad 67\% \]
Results on the Wiki test set.

<table>
<thead>
<tr>
<th></th>
<th>Primary</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LP</td>
<td>LR</td>
</tr>
<tr>
<td>Sparse</td>
<td>64.5</td>
<td>63.7</td>
</tr>
<tr>
<td>MLP</td>
<td>65.2</td>
<td>64.6</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>74.4</td>
<td>72.7</td>
</tr>
</tbody>
</table>

Results on the 20K Leagues out-of-domain set.
## Results

<table>
<thead>
<tr>
<th></th>
<th>Primary</th>
<th></th>
<th>Remote</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LP</td>
<td>LR</td>
<td>LF</td>
<td>LP</td>
</tr>
<tr>
<td>Sparse</td>
<td>64.5</td>
<td>63.7</td>
<td>64.1</td>
<td>19.8</td>
</tr>
<tr>
<td>MLP</td>
<td>65.2</td>
<td>64.6</td>
<td>64.9</td>
<td>23.7</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>74.4</td>
<td>72.7</td>
<td>73.5</td>
<td>47.4</td>
</tr>
</tbody>
</table>

Results on the Wiki test set.

<table>
<thead>
<tr>
<th></th>
<th>Primary</th>
<th></th>
<th>Remote</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LP</td>
<td>LR</td>
<td>LF</td>
<td>LP</td>
</tr>
<tr>
<td>Sparse</td>
<td>59.6</td>
<td>59.9</td>
<td>59.8</td>
<td>22.2</td>
</tr>
<tr>
<td>MLP</td>
<td>62.3</td>
<td>62.6</td>
<td>62.5</td>
<td>20.9</td>
</tr>
<tr>
<td>BiLSTM</td>
<td>68.7</td>
<td>68.5</td>
<td>68.6</td>
<td>38.6</td>
</tr>
</tbody>
</table>

Results on the 20K Leagues out-of-domain set.
Bilexical Graph Approximation

No existing UCCA parsers ⇒ compare to bilexical parsers:
1. Convert UCCA to bilexical dependencies.
2. Train bilexical parsers and apply to test sentences.
3. Reconstruct UCCA graphs and compare with gold standard.

After graduation, Joe moved to Paris

Bilexical DAG approximation.
Baselines

Bilexical DAG parsers:
- DAGParser (Ribeyre et al., 2014): transition-based.

Tree parsers (all transition-based):
- MaltParser (Nivre et al., 2007): bilexical tree parser.
- LSTM Parser (Dyer et al., 2015): bilexical tree parser.
- UPARSE (Maier, 2015): allows non-terminals, discontinuity.

Conversion to trees is just removing remote edges.
### Results

**TUPA_{BiLSTM}** obtains the highest F-scores in all metrics:

<table>
<thead>
<tr>
<th>Model</th>
<th>Primary</th>
<th>Remote</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LP</td>
<td>LR</td>
</tr>
<tr>
<td><strong>TUPA_{Sparse}</strong></td>
<td>64.5</td>
<td>63.7</td>
</tr>
<tr>
<td><strong>TUPA_{MLP}</strong></td>
<td>65.2</td>
<td>64.6</td>
</tr>
<tr>
<td><strong>TUPA_{BiLSTM}</strong></td>
<td>74.4</td>
<td>72.7</td>
</tr>
<tr>
<td>Bilexical DAG</td>
<td></td>
<td>(91)</td>
</tr>
<tr>
<td>DAGParser</td>
<td>61.8</td>
<td>55.8</td>
</tr>
<tr>
<td>TurboParser</td>
<td>57.7</td>
<td>46</td>
</tr>
<tr>
<td>Bilexical tree</td>
<td></td>
<td>(91)</td>
</tr>
<tr>
<td>MaltParser</td>
<td>62.8</td>
<td>57.7</td>
</tr>
<tr>
<td>LSTM Parser</td>
<td>73.2</td>
<td>66.9</td>
</tr>
<tr>
<td>Tree</td>
<td></td>
<td>(100)</td>
</tr>
<tr>
<td>UPARSE</td>
<td>60.9</td>
<td>61.2</td>
</tr>
</tbody>
</table>

Results on the Wiki test set.
## Results

### Similar on out-of-domain test set:

<table>
<thead>
<tr>
<th>Model</th>
<th>Primary LP</th>
<th>Primary LR</th>
<th>Primary LF</th>
<th>Remote LP</th>
<th>Remote LR</th>
<th>Remote LF</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TUPA\textsubscript{Sparse}</strong></td>
<td>59.6</td>
<td>59.9</td>
<td>59.8</td>
<td>22.2</td>
<td>7.7</td>
<td>11.5</td>
</tr>
<tr>
<td><strong>TUPA\textsubscript{MLP}</strong></td>
<td>62.3</td>
<td>62.6</td>
<td>62.5</td>
<td>20.9</td>
<td>6.3</td>
<td>9.7</td>
</tr>
<tr>
<td><strong>TUPA\textsubscript{BiLSTM}</strong></td>
<td>68.7</td>
<td>68.5</td>
<td><strong>68.6</strong></td>
<td>38.6</td>
<td>18.8</td>
<td><strong>25.3</strong></td>
</tr>
<tr>
<td>Bilexical DAG</td>
<td></td>
<td></td>
<td>(91.3)</td>
<td></td>
<td></td>
<td>(43.4)</td>
</tr>
<tr>
<td>DAGParser</td>
<td>56.4</td>
<td>50.6</td>
<td>53.4</td>
<td>–</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TurboParser</td>
<td>50.3</td>
<td>37.7</td>
<td>43.1</td>
<td>100</td>
<td>0.4</td>
<td>0.8</td>
</tr>
<tr>
<td>Bilexical tree</td>
<td></td>
<td></td>
<td>(91.3)</td>
<td></td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>MaltParser</td>
<td>57.8</td>
<td>53</td>
<td>55.3</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>LSTM Parser</td>
<td>66.1</td>
<td>61.1</td>
<td>63.5</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>Tree</td>
<td></td>
<td></td>
<td>(100)</td>
<td></td>
<td></td>
<td>–</td>
</tr>
<tr>
<td>UPARSE</td>
<td>52.7</td>
<td>52.8</td>
<td>52.8</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Results on the 20K Leagues out-of-domain set.
Conclusion

- UCCA’s semantic distinctions require a graph structure including challenging structural properties.
- TUPA is a transition-based parser suitable for UCCA, achieving high accuracy with BiLSTM model.
- Outperforms conversion-based parsing with a variety of strong bilexical DAG and tree baselines.

Code: [https://github.com/danielhers/tupa](https://github.com/danielhers/tupa)
Demo: [https://rebrand.ly/tupa](https://rebrand.ly/tupa)
Future Work

- Beam search, training with exploration.
- More languages (German corpus construction is underway).
- Parsing other schemes, such as AMR.
- Different target representations for conversion.
- Application to text simplification and other tasks.
Thank you
The state of the art in semantic representation.
In Proc. of ACL.
to appear.

UCCAApp: Web-application for syntactic and semantic phrase-based annotation.
to appear.

Lisbon: Evaluating TurboSemanticParser on multiple languages and out-of-domain data.
In Proc. of SemEval, pages 970–973.

Abstract Meaning Representation for sembanking.
In Proc. of the Linguistic Annotation Workshop.


Transition-based dependency parsing with stack long short-term memory.
A primer on neural network models for natural language processing.

Discontinuous incremental shift-reduce parsing.

Dependency grammar and dependency parsing.

Non-projective dependency parsing in expected linear time.

MaltParser: A language-independent system for data-driven dependency parsing.
Natural Language Engineering, 13(02):95–135.

SemEval 2015 task 18: Broad-coverage semantic dependency parsing.
In Proc. of SemEval, pages 915–926.

Alpage: Transition-based semantic graph parsing with syntactic features.
In Proc. of SemEval, pages 97–103.

Conceptual annotations preserve structure across translations: A French-English case study.
In Proc. of S2MT, pages 11–22.

Elements de syntaxe structurale.
## UCCA Corpora

<table>
<thead>
<tr>
<th></th>
<th>Wiki Train</th>
<th>Wiki Dev</th>
<th>Wiki Test</th>
<th>20K Leagues</th>
</tr>
</thead>
<tbody>
<tr>
<td># passages</td>
<td>300</td>
<td>34</td>
<td>33</td>
<td>154</td>
</tr>
<tr>
<td># sentences</td>
<td>4268</td>
<td>454</td>
<td>503</td>
<td>506</td>
</tr>
<tr>
<td># nodes</td>
<td>298,993</td>
<td>33,704</td>
<td>35,718</td>
<td>29,315</td>
</tr>
<tr>
<td>% terminal</td>
<td>42.96</td>
<td>43.54</td>
<td>42.87</td>
<td>42.09</td>
</tr>
<tr>
<td>% non-term.</td>
<td>58.33</td>
<td>57.60</td>
<td>58.35</td>
<td>60.01</td>
</tr>
<tr>
<td>% discont.</td>
<td>0.54</td>
<td>0.53</td>
<td>0.44</td>
<td>0.81</td>
</tr>
<tr>
<td>% reentrant</td>
<td>2.38</td>
<td>1.88</td>
<td>2.15</td>
<td>2.03</td>
</tr>
<tr>
<td># edges</td>
<td>287,914</td>
<td>32,460</td>
<td>34,336</td>
<td>27,749</td>
</tr>
<tr>
<td>% primary</td>
<td>98.25</td>
<td>98.75</td>
<td>98.74</td>
<td>97.73</td>
</tr>
<tr>
<td>% remote</td>
<td>1.75</td>
<td>1.25</td>
<td>1.26</td>
<td>2.27</td>
</tr>
</tbody>
</table>

Average per non-terminal node

| # children       | 1.67       | 1.68     | 1.66      | 1.61        |

Corpus statistics.
Evaluation

Mutual edges between predicted graph $G_p = (V_p, E_p, \ell_p)$ and gold graph $G_g = (V_g, E_g, \ell_g)$, both over terminals $W = \{w_1, \ldots, w_n\}$:

$$M(G_p, G_g) = \left\{ (e_1, e_2) \in E_p \times E_g \mid y(e_1) = y(e_2) \land \ell_p(e_1) = \ell_g(e_2) \right\}$$

The yield $y(e) \subseteq W$ of an edge $e = (u, v)$ in either graph is the set of terminals in $W$ that are descendants of $v$. $\ell$ is the edge label.

Labeled precision, recall and F-score are then defined as:

$$LP = \frac{|M(G_p, G_g)|}{|E_p|}, \quad LR = \frac{|M(G_p, G_g)|}{|E_g|},$$

$$LF = \frac{2 \cdot LP \cdot LR}{LP + LR}.$$

Two variants: one for primary edges, and another for remote edges.