Hebrew Dependency Parsing: Initial Results

Yoav Goldberg  Michael Elhadad

Ben-Gurion University of the Negev

IWPT 2009, Paris
Motivation

- We want a Hebrew Dependency parser
Motivation

- We want a Hebrew Dependency parser
- Initial steps:
  - Know Hebrew
  - Create Hebrew Dependency Treebank
  - Experiment with existing state-of-the-art systems

Next year:
- Do better
Motivation

- We want a Hebrew Dependency parser
- Initial steps:
  - Know Hebrew
  - Create Hebrew Dependency Treebank
  - Experiment with existing state-of-the-art systems
- Next year:
  - Do better
Motivation

- We want a Hebrew Dependency parser
- **Initial steps:**
  - Know Hebrew
  - Create Hebrew Dependency Treebank
  - Experiment with existing state-of-the-art systems
- Next year:
  - Do better
Know Hebrew
Know Hebrew

- Relatively free constituent order
  - Suitable for a dependency based representation

Mostly SVO, but OVS, VSO also possible. Verbal arguments appear before or after the verb.

- went from-Israel to-Paris
- went to-Paris from-Israel
- to-Paris from-Israel went
- to-Paris went from-Israel
Know Hebrew

- Relatively free constituent order
- **Rich morphology**
  - Many word forms
  - Agreement – noun/adj, verb/subj: should help parsing!
Know Hebrew

- Relatively free constituent order
- Rich morphology
- **Agglutination**
  - Many function words are attached to the next token
  - Together with rich morphology ⇒ Very High Ambiguity
  - Leaves of tree not known in advance!
Hebrew Dependency Treebank

- Converted from Hebrew Constituency Treebank (V2)
  - Some heads marked in Treebank
  - For others: (extended) head percolation table from Reut Tsarfaty

- 6220 sentences
- 34 non-projective sentences
Hebrew Dependency Treebank

- Choice of heads
Hebrew Dependency Treebank

- Choice of heads
  - Prepositions are head of PPs

- Root
  - מגיעים
  - ifral
  - arriving
  - from
  - Israel
Hebrew Dependency Treebank

- Choice of heads
  - Prepositions are head of PPs
  - **Relativizers are head of Relative clauses**

```
-Root-
      ↓                ↓                ↓                ↓
     ה the            jish flight    f that    nxth landed
       ↓                    ↓                    ↓
      ו that                נותה landed  אتمול yesterday
```
Choice of heads

- Prepositions are head of PPs
- Relativizers are head of Relative clauses
- **Main verb is head of infinitive verb**
Choice of heads

- Prepositions are head of PPs
- Relativizers are head of Relative clauses
- Main verb is head of infinitive verb
- Coordinators are head of Conjunctions
Choice of heads

- Prepositions are head of PPs
- Relativizers are head of Relative clauses
- Main verb is head of infinitive verb
- Coordinators are head of Conjunctions ← hard for parsers
Dependency labels

- Marked in TBv2
  - OBJ
  - SUBJ
  - COMP
- Trivially added
  - ROOT
  - suffix-inflections
- We are investigating ways of adding more labels
- This work focus on unlabeled dependency parsing.
Experiments
Parameters

Graph vs. Transitions
How important is lexicalization?
Does morphology help?
Parsers

- Transition based: MALTParser (Joakim Nivre)
  - MALT: malt parser, out-of-box feature set
  - MALTARA: malt parser, arabic optimized feature set (should do morphology..)

- Graph based: MST Parser (Ryan Mcdonald)
  - MST1: first order MST parser
  - MST2: second order MST parser
Experimental Setup

- Oracle setting:
  use gold morphology/tagging/segmentation

- Pipeline setting:
  use tagger based morphology/tagging/segmentation
### Results

<table>
<thead>
<tr>
<th>Features</th>
<th>MST1</th>
<th>MST2</th>
<th>MALT</th>
<th>MALT-ARA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Lex</td>
<td>83.60</td>
<td>84.31</td>
<td>80.77</td>
<td>80.32</td>
</tr>
<tr>
<td>Lex 20</td>
<td>82.99</td>
<td>84.52</td>
<td>79.69</td>
<td>79.40</td>
</tr>
<tr>
<td>Lex 100</td>
<td>82.56</td>
<td>83.12</td>
<td>78.66</td>
<td>78.56</td>
</tr>
</tbody>
</table>

**Table:** *oracle* token segmentation and POS-tagging.

<table>
<thead>
<tr>
<th>Features</th>
<th>MST1</th>
<th>MST2</th>
<th>MALT</th>
<th>MALT-ARA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Lex</td>
<td>83.60</td>
<td>84.39</td>
<td>80.77</td>
<td>80.73</td>
</tr>
<tr>
<td>Lex 20</td>
<td>83.60</td>
<td>84.77</td>
<td>79.69</td>
<td>79.84</td>
</tr>
<tr>
<td>Lex 100</td>
<td>83.23</td>
<td>83.80</td>
<td>78.66</td>
<td>78.56</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Features</th>
<th>MST1</th>
<th>MST2</th>
<th>MALT</th>
<th>MALT-ARA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Lex</td>
<td>75.64</td>
<td>76.38</td>
<td>73.03</td>
<td>72.94</td>
</tr>
<tr>
<td>Lex 20</td>
<td>75.48</td>
<td>76.41</td>
<td>72.04</td>
<td>71.88</td>
</tr>
<tr>
<td>Lex 100</td>
<td>74.97</td>
<td>75.49</td>
<td>70.93</td>
<td>70.73</td>
</tr>
</tbody>
</table>

**Table:** *Tagger* token segmentation and POS-tagging.
Results

Best oracle result: 84.77%

Best real result: 76.41%
Results

\[ \text{Mst2} > \text{Mst1} > \text{Malt} \]
Results

\[ \text{Mst2} > \text{Mst1} > \text{MALT} \]

Simply a better model
Results

$$\text{Mst2} > \text{Mst1} > \text{Malt}$$

Partly because of coordination representation
Results

Lexical items appearing > 20 times

∼

all lexical items
Results

With Oracle Morphology

- Morphological features don’t really help
Results

With Tagger Morphology

- Morphological features help MALT a little
- Morphological features *hurt* MST a lot
Where do we go from here?

- We have a Hebrew Dependency Treebank
- Realistic performance still too low
- Current models don’t utilize morphological information well
  - Can we do better?
- Pipeline model hurt performance
  - Can we do parsing, tagging and segmentation jointly?