Conll 2005 – Shared Task
Integrating syntactic parsing and semantic role labeling
Szu-ting Yi & Martha Palmer
Ann Arbor, MI
June, 2005
Motivation

• The bottleneck of the SRL task: parsing
  – With x&p pruning, given different parsers: 12%~18% arguments are lost (sec 22)

• What do we want from syntactic parsing?
  – Correct constituent boundaries
  – Correct tree structures: expressing the dependency between the target verb and its arguments (PATH)

• Our proposed approach:
  – Combine syntactic parsing & argument identification (different cut of the task)
    • Train a new parser on the training data created by merging the Penn Treebank & the PropBank (sec 02-21)
    • Note: The PATH feature is a dominant feature in argument identification.
Data Preparation

• Data preparation steps:
  – Strip off the Penn Treebank function tags
  – 4 sub-labels to represent the PropBank arguments: -AN, -ANC, -AM, -AMC
    • AN: core arguments (A0-5, AA)
    • AM: adjunct-like arguments (AM-*)
    • *C: split arguments

• Example sentence:
  – [ARG0 The luxury auto maker] [ARGM-TMP last year] sold [ARG1 1,214 cars] [ARGM-LOC in the U.S.]
  – (S (NP-AN (DT The) (NN luxury) (NN auto) (NN maker) ) (NP-AM (JJ last) (NN year) ) (VP (VBD sold) (NP-AN (CD 1,214) (NNS cars) ) (PP -AM (IN in) (NP (DT the) (NNP U.S.) ) ) ) ) )
Parsing Experiments

• Base parser: Ratnaparkhi’s (1999) maximum-entropy parser
  – Easy re-implementation
  – Flexibility

• Parser implementation: OpenNLP project (http://sourceforge.net/projects/opennlp/)

• Train 4 parsers:
  – ME: Trained on only the Penn Treebank data (baseline)
  – AN: Penn Treebank + (-AN, -ANC)
  – AM: Penn Treebank + (-AM, -AMC)
  – ANAM: Penn Treebank + (-AN, -ANC, -AM, -AMC)
## Semantic Role Labeling - Results

We trained 3 new parsers: AN, AM & ANAM and we present the best parser (AM) here.

### Test Data: WSJ Sec 23 (5,284 propositions)

<table>
<thead>
<tr>
<th>Parser</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>70.02</td>
<td>61.45</td>
<td>65.45</td>
</tr>
<tr>
<td>AM</td>
<td>78.41</td>
<td>69.60</td>
<td>73.74</td>
</tr>
<tr>
<td>DB</td>
<td>78.87</td>
<td>70.42</td>
<td>74.41</td>
</tr>
</tbody>
</table>

### Test Data: Brown Corpus (7,606 propositions)

<table>
<thead>
<tr>
<th>Parser</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>ME</td>
<td>68.27</td>
<td>58.11</td>
<td>62.78</td>
</tr>
<tr>
<td>AM</td>
<td>72.85</td>
<td>61.18</td>
<td>66.51</td>
</tr>
<tr>
<td>DB</td>
<td>66.36</td>
<td>55.30</td>
<td>60.33</td>
</tr>
</tbody>
</table>

We trained 3 new parsers: AN, AM & ANAM and we present the best parser (AM) here.
Discussion

• Discussion
  – Better adaptability (note: some experiments have been done based on Collins model 2)
  – The anchor verb problem
  – Re-arrange training data
    • AM-MOD, AM-NEG … are not adjuncts
    • Shrink the training trees to keep the core elements only