Beyond the 1-best pipeline

Mark Johnson
Macquarie University
Overview

- 1-best pipeline architectures now standard
  - *Is there value to improving on them?*
- Bidirectional message-passing allows “later” modules to fix “earlier” mistakes
- Also allows training a module without explicit training data
- Building this would be a research challenge
  - Jason Narad’s PhD thesis (his results below)
Conventional 1-best pipeline system

- Each component’s input is the previous component’s output ⇒ cascading errors
- Separate training data required for each component
  - Often only out-of-domain data is available
$n$-best reranking pipeline

- **Multiple complete analyses at each stage**
  - *weights* indicate how good each analysis is
- **Combinatorial ambiguity explosion** $\Rightarrow$ exponential number of analyses required
Packed $n$-best reranking pipeline

- Factor representations into packed values
  - e.g., dynamic programming parsing chart
- Localise ambiguity with packed $n$-best values
  - e.g., Huang’s “cube pruning”
    - represents exponentially many analyses in polynomial space/time
Bidirectional message-passing system

- Bidirectional messages about packed values
  - belief propagation / dual decomposition
  - modules jointly agree on representations
- Backward messages enable learning without module-specific training data
Semantic role labelling

Unlabeled F1

- Baseline
- Hidden
- Oracle

<table>
<thead>
<tr>
<th>Language</th>
<th>Baseline</th>
<th>Hidden</th>
<th>Oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Catalan</td>
<td>74.48</td>
<td>94.01</td>
<td>97.31</td>
</tr>
<tr>
<td>Chinese</td>
<td>68.44</td>
<td>84.53</td>
<td>87.69</td>
</tr>
<tr>
<td>English</td>
<td>80.68</td>
<td>86.31</td>
<td>88.91</td>
</tr>
<tr>
<td>German</td>
<td>60.06</td>
<td>89.00</td>
<td>86.41</td>
</tr>
<tr>
<td>Japanese</td>
<td>58.87</td>
<td>81.25</td>
<td>63.64</td>
</tr>
</tbody>
</table>
Conclusions

- There are more sophisticated architectures beyond the 1-best pipeline
  - Bidirectional message-passing doesn’t need training data for all modules
- More complex architectures require more coordination between modules
  - Additional complexity doesn’t always “pay off”
  - “Sweet spot” depends on intended applications