Multilingual Dependency Parsing

Keith Hall
JHU CLSP
Outline

- Two-stage Non-projective Dependency Parser (tree structure = unlabeled dependencies)
  - algorithm
  - models
  - training
- Empirical Results on 8 languages
- Applications
  - Speech Reconstruction
  - Low-density language parsing
Parser Reranking

- Rerank trees output by parser
- Based on discriminative modeling techniques
- Often, modeled as “best vs. rest”
  - Constituency Structures (Collins ICML00, Charniak & Johnson ACL05)
  - LFG (Riezler et al. ACL02)
Dependency Reranker

- **McDonald** *(ACL05)*
  - Used k-best Eisner algorithm to generate training data
  - Trained large margin classifier
  - Multi-label classification problem (not best vs. rest)
  - Also presented edge-factored models
Reranking and OT

c.f. Stevenson & Smolensky
Ch. 19: “Optimality in Sentence Processing”
Toward a calculus of the mind/brain: Neural network theory, optimality, and universal grammar

(9) John put the candy on

a. the table.

b. # the table into his mouth.

<table>
<thead>
<tr>
<th>PP Argument/NP Adjunct</th>
<th>Ob-HD</th>
<th>Assign-θ</th>
<th>Locality</th>
</tr>
</thead>
<tbody>
<tr>
<td>John put the candy on</td>
<td>VP</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V'</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ the table. ]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>[ # the table into[ his</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mouth. ]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

John put the
candy on

the table.

*NP

*ε
Two Stage Parser

- k-best MST algorithm
  - Discriminatively trained
  - Efficient $O(n^2)$
- Reranker
  - Discriminatively trained
- Fast: $O(k \cdot n^2)$
  - Training is slow
MST Parsing

- Maximum Spanning Tree
  best dependency tree
  natural similarity

- $s(T) = \sum_i s(g_i \rightarrow w_i)$

- Edge-factored constraint
  $s(a \rightarrow b)$ is independent of $s(c \rightarrow b)$

- $s()$ can use all other information from the input
  - linear order  $s(a \rightarrow b)$ (a comes before b)
  - neighboring nodes  $s(a \rightarrow b)$ (c follows child b)
k-best MST

- Efficient version of Edmonds/Chu-Liu algorithm (Edmonds algorithm used by McDonald et al. EMNLP05)
- Based on Camerini et al. 1980
- Finding “next” best tree is key to efficiency
- General setup is based on constrained graph
k-best MST setup
k-best MST setup

\[ G = \{E,V\} = \]

Graph

ROOT

a

b

c
k-best MST setup

\[ G = \{E, V\} = \]

**Graph**

**Branching**

\[ Y = \]

![Diagram of a graph with nodes a, b, and c, and edges connecting them. A branching with nodes a and c, and edge b connecting them.]
k-best MST setup

Graph

Branching

Exclusion Set

\[ G = \{E, V\} = \]

\[ Y = \]

\[ Z = \]
k-best MST setup

Graph

\[ G = \{E,V\} = \]

Branching

\[ Y = \]

Exclusion Set

\[ Z = \]

Constrained Graph

\[ G_{Y,Z} = \]
k-best MST setup

- Constrained graph $G_{Y,Z}$
  - $Y$ must be part of MST of $G_{Y,Z}$
  - $Z$ cannot be part of MST ...

- iff $A$ is an MST of $G$
  and $Y \subseteq A \subseteq E - Z$
  then $A$ is an MST of $G_{Y,Z}$

- Solving constrained problem leads to unconstrained solution (used for k-best)
MST example

M (graph)

Root

C (branching)

a → b → c
MST example

M (graph)

C (branching)

Root

a

b

c

a
MST example

M (graph)  C (branching)

Root

b

a  c

a
MST example

M (graph)

Root

a

b

c

C (branching)

b

a
MST example

\[ M \] (graph)

Root

\[ b \]

\[ a \rightarrow c \]

\[ C \] (branching)

\[ a \rightarrow \]

\[ b \rightarrow a \]
MST example

M (graph)

Root

- b
- a
- c

C (branching)

- d
- b
- a
MST example

M (graph)

Root

d  c

C (branching)

b  a

d

d
Next Best

- Given best tree $A$ and graph $G_{Y,Z}$
- An edge $e$ exists in $A$:
  - when removed from $A$ and from the graph (i.e., $G_{Y,\{Z \cup e\}}$)
    best tree of $G_{Y,\{Z \cup e\}}$ is the next best of $G_{Y,Z}$
- Branching $C$ used to find $e$
Third Best

- $G_{Y\cup \{Z \cup e\}}$ - graph without e
  - MST is next best of $G_{Y,Z}$
  - What about the “next” of that graph

- $G_{\{Y \cup e\},Z}$ - graph with e
  - MST is best of $G_{Y,Z}$
  - What about the “next” of this graph

- One of these is the third best
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set

```
... 1 2 3 ...
```

```
1
2
3
...
```
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set

\[ \text{i}^{\text{th}} \text{ best} \]

next of \( G_{Y, \{Z \cup e\}} \)

next of \( G_{\{Y \cup e\}, Z} \)
k-best MST

- Priority queue of trees
- Use constrained graph of get partition of solution set
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)

```
ROOT
share
two
house
a
almost
devoid
of
furniture
```
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats
  (from CoNLL data)
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats
  (from CoNLL data)

ROOT
share
two
house
a
devoid
almost
of
furniture

two
share
a
devoid
almost

almost
devoid
of

furniture

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT

ROOT
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)

```
two share a house almost devoid of furniture.
```
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)

ROOT

two
share
house
devoid
almost
of
furniture

almost
devoid
of
furniture
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)

ROOT
two
share
house
almost
devoid
of
furniture
.devoid

ROOT

. // End of sentence
Edge-factored Models

- Node-Type
- Branch
- Distance
- Inside/Outside
- Edge
- “Extra” feats (from CoNLL data)
## Training Edge-factored Models

<table>
<thead>
<tr>
<th>Child</th>
<th>Parent</th>
<th>Correct</th>
<th>$f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>two</td>
<td>ROOT</td>
<td>0</td>
<td>RtDt, Y,N,5,</td>
</tr>
<tr>
<td>two</td>
<td>share</td>
<td>1</td>
<td>VbDt, Y,N,5,</td>
</tr>
<tr>
<td>two</td>
<td>a</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>share</td>
<td>ROOT</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>share</td>
<td>two</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>share</td>
<td>a</td>
<td>0</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- $n^2$ examples/sentence
- Large training sets
Reranking Models

- MST score
- Sibling
- Valency (count) & Subcategorization (frames)
- Ancestor - grandparent, great-grandparent
- Edge - neighbors or completed subtrees
- Branch Factor (ratio and boolean)
- Depth - height of tree
- Heavy - Number of dominated nodes per type
  (Charniak & Johnson ACL05)
Training Reranker

- $k \times \text{num. training sentences}$
- Jack-knife training
  - use entire training set
  - no bias towards training

<table>
<thead>
<tr>
<th>Sent/Tree</th>
<th>Accuracy</th>
<th>$f$</th>
</tr>
</thead>
<tbody>
<tr>
<td>sent 1, tree 1</td>
<td>89%</td>
<td>67,R,</td>
</tr>
<tr>
<td>sent 1, tree 2</td>
<td>85%</td>
<td>75,R,</td>
</tr>
<tr>
<td>sent 1, tree 3</td>
<td>55%</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>sent 1, tree k</td>
<td>53%</td>
<td></td>
</tr>
<tr>
<td>sent 2, tree 1</td>
<td>88%</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>sent 2, tree k</td>
<td>50%</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
**Common Modeling Paradigm**

<table>
<thead>
<tr>
<th>Example</th>
<th>Score</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>set 1, ex. 1</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>set 1, ex. 2</td>
<td>90</td>
<td></td>
</tr>
<tr>
<td>set 1, ex. 3</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>set 2, ex. 1</td>
<td>88</td>
<td></td>
</tr>
<tr>
<td>set 2, ex. 1</td>
<td>87</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

**Example**

<table>
<thead>
<tr>
<th>Example</th>
<th>Score</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>set 1, ex. 1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>set 1, ex. 2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>set 1, ex. 3</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>set 2, ex. 1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>set 2, ex. 1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

- **Best versus the rest (reranking)**
  - Collins ICML00

- edge-factored and reranking models use the same optimization routine
Inference Functions

- MST Parser Scores:

\[ s(w_i, w_j) = p_\theta(y_i \in Y_+ | Y_e) \]

\[ y_i \in Y_+ \iff e(w_i, w_j) = 1 \]

- Reranker objective:

\[ \arg \max_i p_\theta(y_i \in Y_+ | Y_e) \]
MaxEnt Training

- Given conditional distribution

\[
p_\theta(y_i \in Y_+|Y_e)
\]

- Match observations

\[
E_{p_\theta}(f) = E_{\hat{p}}(f)
\]

- Log-linear form

\[
p_\theta(y_i \in Y_+|Y_e) = \frac{1}{Z} \exp^{\theta \cdot f(y_i)}
\]

\[
Z = \sum_{j \in Y_e, j \neq i} \exp^{\theta \cdot f(y_j)}
\]
Too Much Data

- edge-factored model training can result in large numbers of “examples”

- Bagging approach (Breiman):
  - Sub-sample (with replacement) n sets
  - At inference combine n classifiers
  - Good for high-variance data-sets

- Our approach:
  - Partition data into n disjoint subsets
  - At inference combine n classifiers
Empirical Evaluation

- 7 Languages from CoNLL-X 2006 shared task on dependency parsing
- + English
- No language-specific training (other than treebanks)
- Some morphological analysis provided (coarse POS-tags are most useful)
- Preliminary reranking results on 3 languages
### k-best Oracle Rates

<table>
<thead>
<tr>
<th>Language</th>
<th>Best Reported</th>
<th>$k = 1$</th>
<th>$k = 10$</th>
<th>$k = 50$</th>
<th>$k = 100$</th>
<th>$k = 500$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>79.34</td>
<td>77.61</td>
<td>80.41</td>
<td>81.86</td>
<td>82.65</td>
<td>84.15</td>
</tr>
<tr>
<td>Czech</td>
<td>87.30</td>
<td>84.08</td>
<td>88.92</td>
<td>91.12</td>
<td>91.94</td>
<td>93.62</td>
</tr>
<tr>
<td>Danish</td>
<td>90.58</td>
<td>89.12</td>
<td>92.89</td>
<td>94.79</td>
<td>95.29</td>
<td>96.59</td>
</tr>
<tr>
<td>Dutch</td>
<td>83.57</td>
<td>81.05</td>
<td>87.43</td>
<td>90.30</td>
<td>91.28</td>
<td>93.12</td>
</tr>
<tr>
<td>English</td>
<td>92.22</td>
<td>88.52</td>
<td>92.15</td>
<td>93.95</td>
<td>94.49</td>
<td>95.76</td>
</tr>
<tr>
<td>German</td>
<td>90.38</td>
<td>87.48</td>
<td>92.27</td>
<td>94.15</td>
<td>94.93</td>
<td>96.01</td>
</tr>
<tr>
<td>Portuguese</td>
<td>91.36</td>
<td>89.38</td>
<td>92.72</td>
<td>94.44</td>
<td>95.05</td>
<td>96.17</td>
</tr>
<tr>
<td>Swedish</td>
<td>89.54</td>
<td>86.50</td>
<td>91.20</td>
<td>93.37</td>
<td>93.83</td>
<td>95.42</td>
</tr>
<tr>
<td>Language</td>
<td>Best Reported</td>
<td>Accuracy $k$-best MST and Reranker</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
<td>-----------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Arabic</td>
<td>79.34</td>
<td>1-best 77.61</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>50-best 77.92</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>100-best 78.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>500-best 77.83</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Czech</td>
<td>87.30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Danish</td>
<td>90.58</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dutch</td>
<td>83.57</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>English</td>
<td>92.22</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>German</td>
<td>90.38</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Portuguese</td>
<td>91.36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Swedish</td>
<td>89.54</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1Table is incomplete at time of submission. Final results will be available in time for final publication.
Review

- k-best MST parser
- Unified modeling approach
  - Trained via MaxEnt
  - Training for large datasets - Bagging
- Allows for arbitrary features
  - edge-factored as preprocessor
  - tree models allow for rich set of features
Applications

- **Current work**
  - Speech Reconstruction
  - Low-density Language Parsing

- **Future work**
  - Incremental MST Parsing
  - MST MT
Reconstruction

- w/ Erin Fitzgerald and Fred Jelinek
- Finding data for speech reconstruction
- Train parser(s) on representative data
  - One parser trained on “fluent” data
  - One parser trained on “disfluent” data (superset of “fluent” data)
- Preferred parser classification
  - Disfluent parser should accept disfluent data
  - Clean parser should dislike disfluent data
Low-density Language

- w/ Brock Pytlik and David Yarowsky
- Language with few resources (e.g., treebanks, etc.) but have some bilingual dictionaries
- Dictionary based techniques to generate glosses (possibly using “bridge” languages)
- Parsing glosses with lexical dependency parser
  - Trained on order independent features
  - Can recover some semantic relationship
Incremental MST

- Left-to-right building of max spanning tree
- Utilize similar properties as k-best
- May not be polynomial
MST MT

- Structural transfer based on MT by parsing
- Alignment/transduction model used to construct decoding graph
- MST algorithm used as decoder
Conclusion

- Two-stage k-best MST parser
  - Common training paradigm
  - Two related feature sets (reranker allows anything)
  - Competitive results (compared to CoNLL best)
  - k-best output (actual reranking)
- Flexibility (applications)
  - Takes advantage of lexically trained parser (no morphological or syntactic features)
  - Robust Multilingual Use