Inspecting the Structural Biases of Dependency Parsing Algorithms

Yoav Goldberg and Michael Elhadad

Ben Gurion University

ISCOL 2010, TAU
Dependency Parsing

Input: a sentence
"the input is a sentence"

Output: dependency tree

We'll be using this notation

```
the input is a sentence
```

```
the
```

```
input
```

```
sentence
```

```
is
```

```
a
```
Parsing Approaches

Graph Based
- global inference
  - expensive! $O(n^3)$++
- edge factored features
  (add some more with high cost)
Parsing Approaches

Graph Based
- global inference
- expensive: \( O(n^3)^{++} \)
- edge factored features

Transition Based
- shift-reduce variants
- many local greedy actions
- left to right
- rich features
- fast! \( O(n) \)
Parsing Approaches

- Graph Based
  - global inference
  - expensive! $O(n^3)^{++}$
  - edge factored features

- Transition Based
  - shift reduce variants
  - many local greedy actions
  - left to right
  - rich features
  - fast! $O(n)$

Hybrids

- Voting
- Stacking
- Blending
Parsing Approaches

Graph Based
- Global inference
- Expensive: $O(n^3)^4$
- Edge factored features

Easy First
NEW!
- Today.

Hybrids
- Voting
- Stacking
- Blending

Transition Based
- Shift reduce variants
- Many local greedy features
- Left to Right
- Rich features
- Fast! $O(n)$
Easy First Parsing

New!

greedy bottom up parser

left to right $\rightarrow$ easy before hard

tast! $O(n\log n)$

less error propagation

parser learns what's easy for it
"the boy ate the salad with the shiny silver fork"
Motivation

"the boy ate the salad with the shiny silver fork"

Graph Based $\rightarrow$ each edge scored separately

Not enough information to resolve ambiguity!
Motivation

"the boy ate the salad with the shining silver fork"

Transition Based

? ate salad "with the shining silver fork"

the the boy the

parser sees up to here

not enough info to resolve ambiguity!
Motivation

"the boy ate the salad with the shiny silver fork"

Transition Based

ate salad

boy the

the

but this is easy to parse

with the shiny silver fork

not enough info to resolve ambiguity!

parser sees up to here
"the boy ate the salad with the shiny silver fork"

Easy First

ate salad with

the shiny silver fork
Motivation

"the boy ate the salad with the shiny silver fork"

Easy First

All needed information is available!
Parsers

MST
  ↓
Ryan McDonald
  Graph Based
    (first order)

MALT
  ↓
Joahim Nivre
  Transition Based
    (arc-eager, poly. SVM classifier)

Easy First
  ↓
This work
<table>
<thead>
<tr>
<th></th>
<th>Unlabeled accuracy</th>
<th>Root accuracy</th>
<th>Complete Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malt</td>
<td>88.36</td>
<td>87.04</td>
<td>34.16</td>
</tr>
<tr>
<td>MST</td>
<td>90.05</td>
<td>93.95</td>
<td>34.64</td>
</tr>
<tr>
<td>Easy First</td>
<td>89.70</td>
<td>91.50</td>
<td>37.5</td>
</tr>
</tbody>
</table>
Our Parses are Different
Parser Combination: Oracle

Parser -> Parser2 -> Parsek

gold -> choose best parse -> parse1
**Parser Combination: Oracle**

<table>
<thead>
<tr>
<th>Combination</th>
<th>Accuracy</th>
<th>Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malt + Mst</td>
<td>92.29</td>
<td>44.03</td>
</tr>
<tr>
<td>Easy + Malt</td>
<td>92.19</td>
<td>45.48</td>
</tr>
<tr>
<td>Easy + Mst</td>
<td>92.53</td>
<td>44.41</td>
</tr>
<tr>
<td>Easy + Malt + Mst</td>
<td><strong>93.54</strong></td>
<td><strong>40.79</strong></td>
</tr>
</tbody>
</table>

**WSJ**
Parser Combination: REAL

Malt  MST  EasyFirst

Sagae and Lavie 2006

90.8

for CoNLL English

(Highest of all participants!)
We can build many accurate parsers

- MALT, MST, CoNLL 2007, EASYFIRST, Liang and Kenji’s.

Parser combinations work

⇒ every parser has its strong points

Different parsers behave differently
McDonald and Nivre 2007:

“Characterize the Errors of Data-Driven Dependency Parsing Models”

- Focus on single-edge errors
  - MST better for long edges, MALT better for short
  - MST better near root, MALT better away from root
  - MALT better at nouns and pronouns, MST better at others
- …but all these differences are very small
we do something a bit different
Assumptions

- Parsers fail in predictable ways
- those can be analyzed
- analysis should be done by inspecting trends rather than individual decisions
Note: We do not do error analysis

- Error analysis is **complicated**
  - one error can yield another / hide another

- Error analysis is **local** to one tree
  - many factors may be involved in that single error

we are aiming at more global trends
Structural preferences
Structural preferences

for a given language+syntactic theory

- Some structures are more common than others
  - (think Right Branching for English)

- Some structures are very rare
  - (think non-projectivity, OSV constituent order)
Structural preferences

parsers also exhibit structural preferences

▶ some are explicit / by design
  ▶ e.g. projectivity

▶ some are implicit, stem from
  ▶ features
  ▶ modeling
  ▶ data
  ▶ interactions
  ▶ and other stuff

These trends are interesting!
Structural Bias
“The difference between the structural preferences of two languages”

For us:

*Which structures tend to occur more in language than in parser?*
Bias vs. Error

related, but not the same

*Parser X makes many PP attachment errors*
  ▶ claim about error pattern

*Parser X tends to attach PPs low, while language Y tends to attach them high*
  ▶ claim about structural bias (and also about errors)

*Parser X can never produce structure Y*
  ▶ claim about structural bias
Formulating Structural Bias

“given a tree, can we say where it came from?”
Formulating Structural Bias

“given two trees of the same sentence, can we tell which parser produced each parse?”
Formulating Structural Bias

“which parser produced which tree?”

any predictor that can help us answer this question is an indicator of structural bias

uncovering structural bias = searching for good predictors
Method

- start with two sets of parses for same set of sentences
- look for predictors that allow us to distinguish between trees in each group
Our Predictors

- all possible subtrees
- always encode:
  - part of speech
  - relations
  - direction
- can encode also:
  - lexical items
  - distance to parent
boosting with subtree features

very briefly:

- input: two sets of constituency trees
- while not done:
  - choose a subtree that classifies most trees correctly
  - re-weight trees based on errors
- output: weighted subtrees (= linear classifier)
Setup

Gold trees
Parsed trees

Train

Validation

KSM 2004

Weighted Subtrees = Classifier

Ignore Weights

Subtrees

Rescore (Count based)

Bias Predictors
conversion to constituency

mandatory information at node label
optional information as leaves
Experiments

Analyzed Parsers

- Malt Eager
- Malt Standard
- Mst 1
- Mst 2

Data

- WSJ (converted using Johansson and Nugues)
- splits: parse-train (15-18), boost-train (10-11), boost-val (4-7)
- gold pos-tags
Q: Are the parsers biased with respect to English?
A: Yes

<table>
<thead>
<tr>
<th>Parser</th>
<th>Train Accuracy</th>
<th>Val Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>MST1</td>
<td>65.4</td>
<td>57.8</td>
</tr>
<tr>
<td>MST2</td>
<td>62.8</td>
<td>56.6</td>
</tr>
<tr>
<td>MALT E</td>
<td>69.2</td>
<td>65.3</td>
</tr>
<tr>
<td>MALT S</td>
<td>65.1</td>
<td>60.1</td>
</tr>
</tbody>
</table>

Table: Distinguishing parser output from gold-trees based on structural information
Qualitative Results (teasers)

Over-produced by ArcEager:

\[
\text{ROOT} \rightarrow " \quad \text{ROOT} \rightarrow \text{DT} \quad \text{ROOT} \rightarrow \text{WP} \\
\text{ROOT} \rightarrow \text{VBD} \quad \text{VBD}
\]

(we knew it’s bad at root, now we know how!)
Qualitative Results (teasers)

Over-produced by ArcEager and ArcStandard

\[ \rightarrow \text{VBD} \underset{9}{\rightarrow} \text{VBD} \]

\[ \rightarrow \text{VBD} \underset{5}{\rightarrow} \text{VBD} \]

ROOT \rightarrow \text{VBZ} \rightarrow \text{VBZ}

(prefer first verb above second one: because of left-to-right processing?)
Qualitative Results (teasers)

Over-produced by MST1

(independence assumption failing)
Qualitative Results (teasers)

Under-produced by MST1 and MST2

(\text{hard time in coordinating “heavy” NPs: due to pos-in-between feature?})
Qualitative Results (teasers)

Software available

Try with your language / parser
To Conclude

- understanding HOW parsers behave and WHY is important
  - we should do more of that

- we defined structural bias as way of characterizing behaviour

- we presented an algorithm for uncovering structural bias

- applied to English with interesting results