Inspecting the Structural Biases of Dependency Parsing Algorithms

Yoav Goldberg and **Michael Elhadad**

Ben Gurion University

ISCOL 2010, TAU

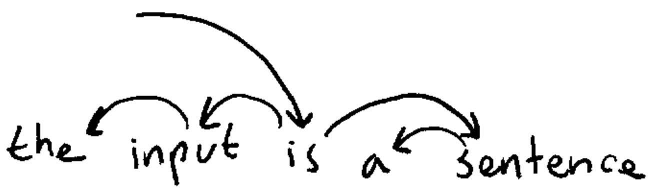
Dependency Parsing

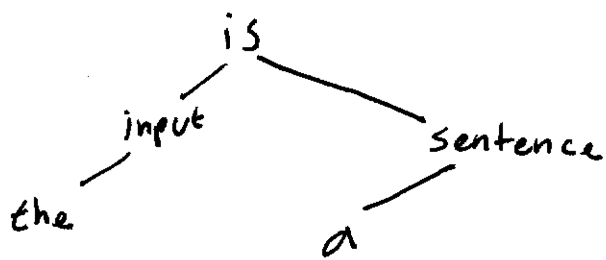
input: a sentence

"the input is a sentence"

output: dependency tree

We'll be using this notation





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! $O(n^3)^{++}$
- edge factored features

Easy First

NEW!

Today.

Transition Based

- Shift reduce variants
- Many local greedy features
- Left to Right
- Rich features
- fast! $O(n)$

Hybrids

- Voting
- Stacking
- Blending

Easy First Parsing

New!

greedy bottom up parser

~~left to right~~ → easy before hard

fast! $O(n \log n)$

less error propagation

parser learns what's easy for it

Motivation

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

Motivation

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

Graph Based → each edge scored seperately

ate $\xrightarrow{?}$ with   salad $\xrightarrow{?}$ with

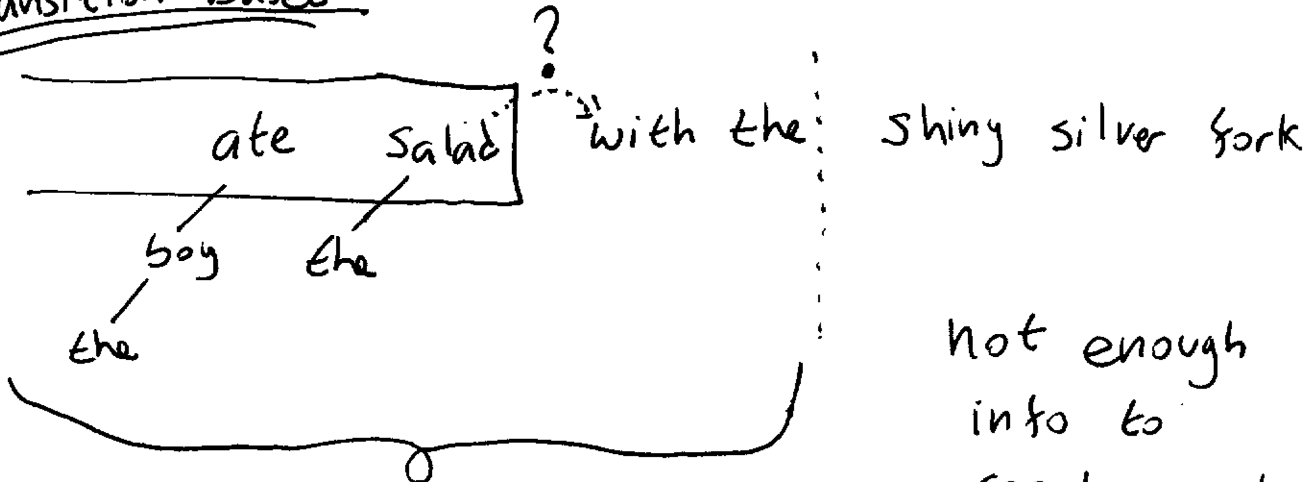
with $\xrightarrow{?}$ fork

Not enough information to resolve ambiguity!

Motivation

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

Transition Based



parser sees up to here

not enough info to resolve ambiguity!

Motivation

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

but this is easy to parse

Transition Based

?

ate salad | with the | shiny silver fork

boy the

the

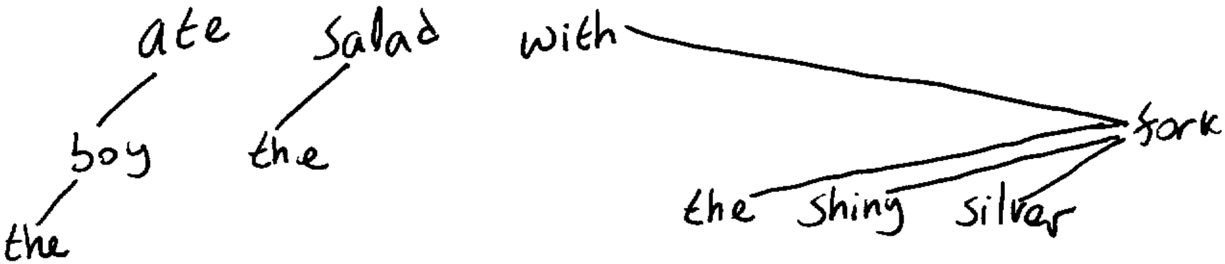
parser sees up to here

not enough info to resolve ambiguity!

Motivation

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

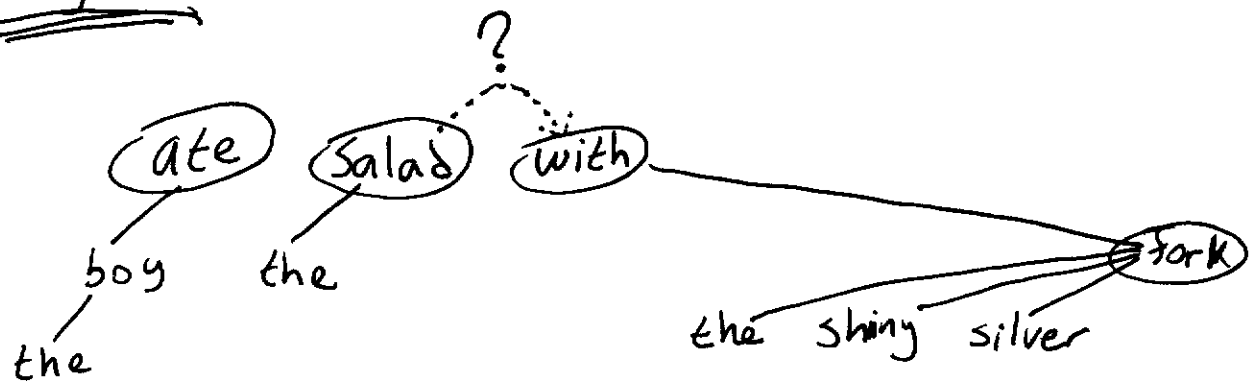
Easy First



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

Joakim Nivre

Transition Based

(arc-eager,
poly. SVM
classifier)

Easy First

This work

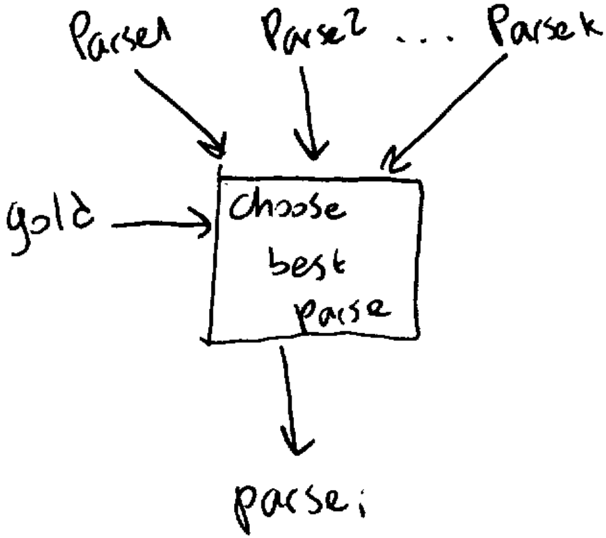
Results

WSJ

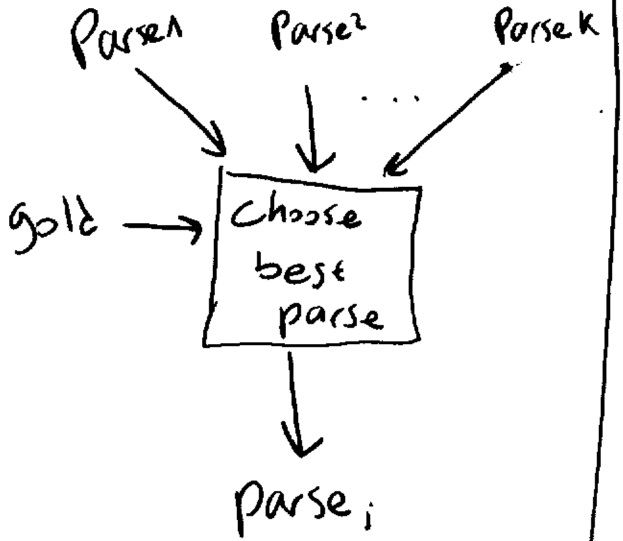
	unlabelet accuracy	root accuracy	complete Sentence
Malt	88.36	87.04	34.16
MST	90.05	93.95	34.64
EasyFirst	89.70	91.50	37.5

Our Parses are Different

Parser Combination: Oracle



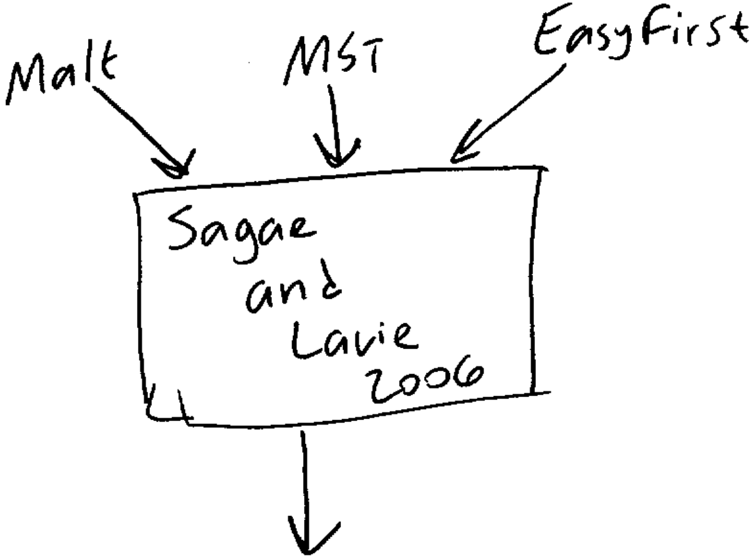
Parser Combination: Oracle



	accuracy	complete
Malt+MST	92.29	44.03
Easy+Malt	92.19	45.48
Easy+Mst	92.53	44.41
Easy+Malt+MST	93.54	49.79

WSJ

Parser Combination: REAL



90.8

for CoNLL English

(Highest of all participants!)

. . . up until now

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

Previously

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

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)

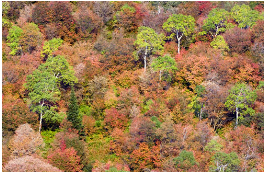
Parser X can never produce structure Y

- claim about structural bias

Formulating Structural Bias

"given a tree, can we say where it came from?"

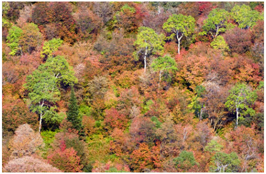
?



Formulating Structural Bias

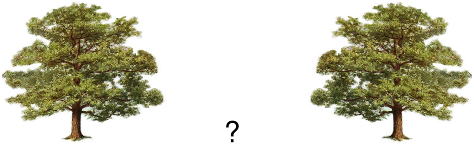
"given two trees of the same sentence, can we tell which parser produced each parse?"

?



Formulating Structural Bias

"which parser produced which tree?"



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

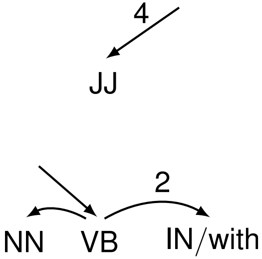
uncovering structural bias = searching for good predictors

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



Search Procedure

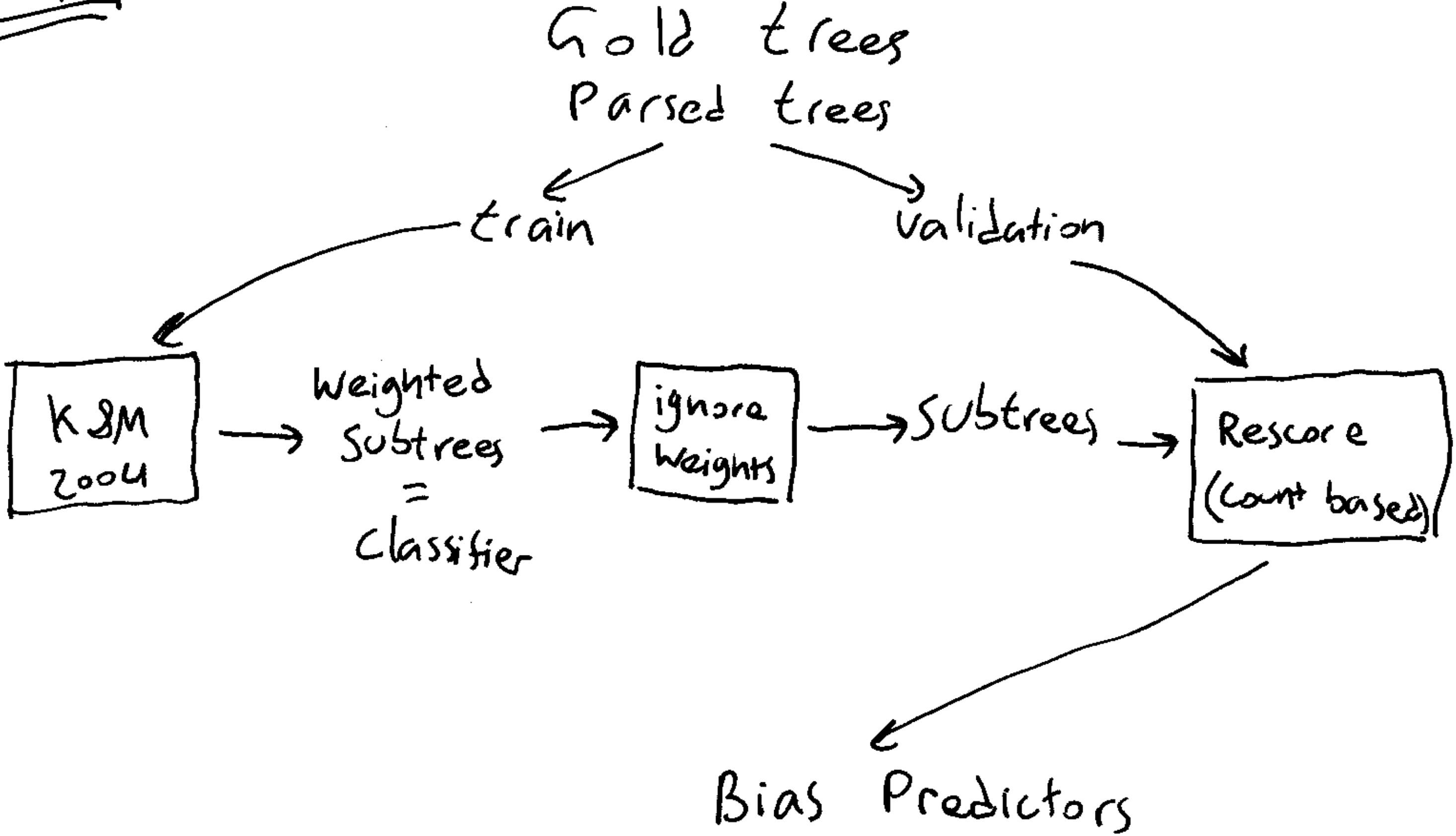
boosting with subtree features

algorithm by Kudo and Matsumoto 2004.

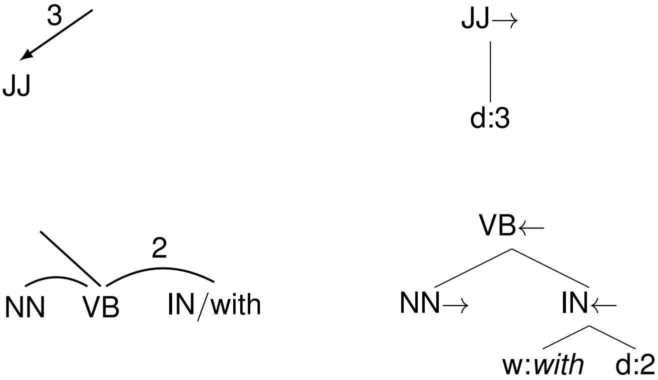
very briefly:

- **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
K&M
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

Quantitative Results

Q: Are the parsers biased with respect to English?
A: Yes

Parser	Train Accuracy	Val Accuracy
MST1	65.4	57.8
MST2	62.8	56.6
MALTE	69.2	65.3
MALTS	65.1	60.1

Table: Distinguishing parser output from gold-trees based on structural information

Qualitative Results (teasers)

Over-produced by ArcEager:

ROOT$\rightarrow$“ ROOT$\rightarrow$DT ROOT$\rightarrow$WP

ROOT VBD VBD

(we knew it's bad at root, now we know how!)

Qualitative Results (teasers)

Over-produced by ArcEager and ArcStandard

$$\rightarrow\text{VBD} \xrightarrow[9+]{} \text{VBD}$$

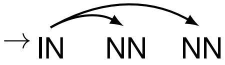
$$\rightarrow\text{VBD} \xrightarrow[5-7]{} \text{VBD}$$

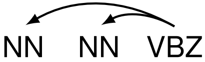
$$\text{ROOT}\rightarrow\text{VBZ}\rightarrow\text{VBZ}$$

(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

$\rightarrow$ NN IN CC NN

(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