Discontinuous treebank annotation using LCFRS

Andreas van Cranenburgh

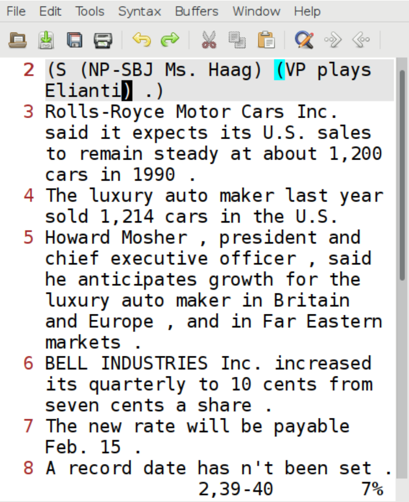
Heinrich Heine Universität Düsseldorf

February 21, 2018

TreeGrasp meeting #1, Düsseldorf

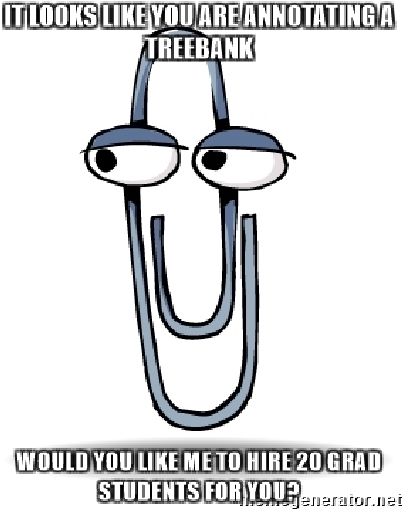
Annotating. 2 down, 40,000 to go . . .

Annotating. 2 down, 40,000 to go . . .



File Edit Tools Syntax Buffers Window Help

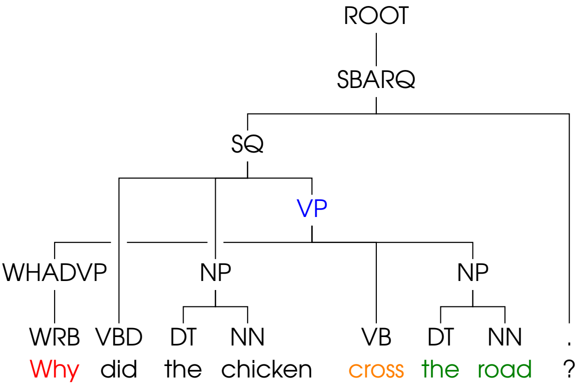
```
2 (S (NP-SBJ Ms. Haag) (VP plays
  Elianti) .)
3 Rolls-Royce Motor Cars Inc.
  said it expects its U.S. sales
  to remain steady at about 1,200
  cars in 1990 .
4 The luxury auto maker last year
  sold 1,214 cars in the U.S.
5 Howard Mosher , president and
  chief executive officer , said
  he anticipates growth for the
  luxury auto maker in Britain
  and Europe , and in Far Eastern
  markets .
6 BELL INDUSTRIES Inc. increased
  its quarterly to 10 cents from
  seven cents a share .
7 The new rate will be payable
  Feb. 15 .
8 A record date has n't been set .
                 2,39-40        7%
```



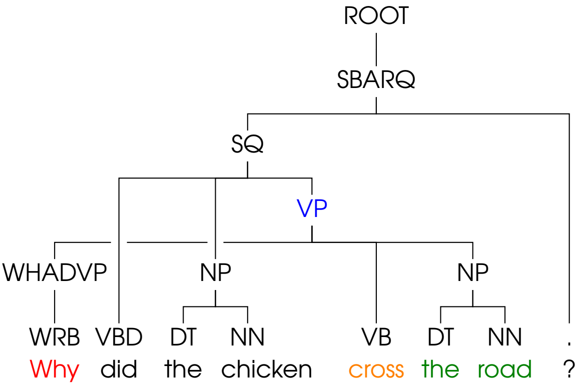
Goals

- Minimize total cost of annotation
  - What is the effect of immediate grammar re-training?
- Annotate literary texts
- Explore other annotation tasks / schemes (RRG?)

Discontinuity with LCFRS



Discontinuity with LCFRS

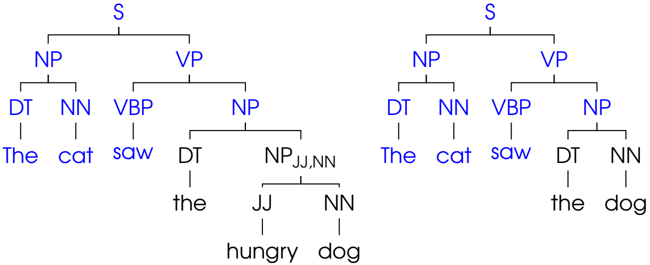


Linear Context-Free Rewriting System (LCFRS)

$$VP_2(a, bc) \rightarrow WHADVP(a)\ VB(b)\ NP(c)$$

Capture constituency + predicate-argument structure

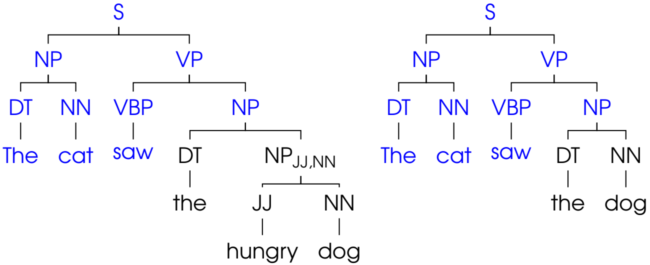
Double-DOP: exploiting common tree fragments



Capture arbitrary word/constituent co-occurrences.

Sangati & Zuidema (2011). Accurate parsing w/compact TSGs: Double-DOP

Double-DOP: exploiting common tree fragments



Capture arbitrary word/constituent co-occurrences.

- Extract fragments that occur at least twice in treebank
- For every pair of trees, extract maximal overlapping fragments
- Fragments can be used as Tree-Substitution Grammar

Sangati & Zuidema (2011). Accurate parsing w/compact TSGs: Double-DOP

Improving parsers with data

Raw text is cheap,
annotation is costly

Unsupervised / semi-supervised: word co-occurrences provide some distributional syntactic information, but limited.

Supervised: Very labor intensive, requires very special set of skills, costly, boring, tedious, etc.

Active Learning: Reduce work load without compromising on annotation quality / detail
$\Rightarrow$ this talk

Actual treebank annotation practice

Manual correction of automatic parses in GUI

PTB: Deterministic parser (Marcus et al 1993, §4.1). Produces only 1 analysis, only provides bracketings it is confident about.

Tiger: Brants et al (2004, §3)

- Interactive annotation with Cascaded Markov Model; advantage: responds to user feedback.
- LFG parser, non-interactive post-editing/disambiguation; advantage: always syntactically consistent.

Efficient annotation

Interactivity :

- Semi-automatic annotation: parser suggests candidates
- Interactive disambiguation: help annotator identify correct analysis

Active Learning :

- Prioritization: Annotate sentences in order that minimizes required user interaction $\Rightarrow$ learning converges faster
- Incremental parser training: further automatic parses *immediately* improve from annotation feedback

Active Learning

1. Select data point that model expects to yield the most improvement. (Training Utility Value)
2. Expert annotates data point.
3. Re-train the model.
4. Repeat.

i.e., machine *teaching* instead of machine learning
(`http://prodi.gy`)

Provides substantial annotation speedup:
e.g., 80 % reduction in annotation time
(Baldridge & Osborne, EMNLP 2004)

Settles (2010), Active learning literature survey.
`http://burrsettles.com/pub/settles.activelearning.pdf`

Ranking sentences I: entropy

Intuition

Disambiguation is hard when a sentence has many analyses with similar probabilities,
$\Rightarrow$ entropy as Training Utility Value (TUV);
Maximizes information gain

1. Collect n-best parse trees
   with probabilities p_i for a sentence
2. Take entropy of probability distribution $p_1 \dots p_n$:
   $-\sum_i p_i \log p_i$
3. Normalize by number of parse trees n:
   TUV(sent) = $\frac{1}{\log n} \cdot -\sum_i p_i \log p_i$

Hwa (CL journal, 2004) Sample Selection for Statistical Parsing.

Ranking sentences II: clustering

Cluster syntactically similar sentences

- Similarity metric: common tree fragments
- Clustering method: K-Means, with k s.t. clusters consist of about 10 sentences

Combine with entropy ranking by first clustering, then ordering the clusters by mean entropy.

- Cluster 1: Once upon a time … etc.
- Cluster 2: …lived happily ever after. etc.
- etc.

Tang et al (ACL 2002), Active Learning for Stat. Nat. Lang. Parsing

Selecting from n-best list: decision tree

Reduce n-best list to a decision tree
of `discriminants'

- Entropy-based decision tree
- Features: presence of bracketings
- Leaves: n-best trees
- Use probabilities: lower prob. $\Rightarrow$ longer path
- Pruning: discard trees with $p < 1/n$

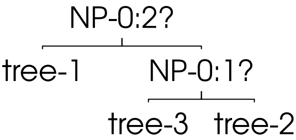
Osborne & Baldridge (EMNLP 2004),
Ensemble-based Active Learning for Parse Selection

Selecting from n-best list: decision tree

Reduce *n*-best list to a decision tree
of 'discriminants'

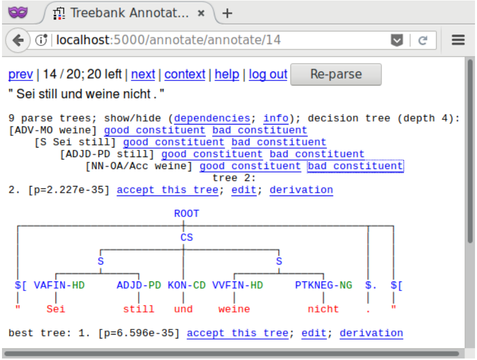
- Entropy-based decision tree
- Features: presence of bracketings
- Leaves: n-best trees
- Use probabilities: lower prob. $\Rightarrow$ longer path
- Pruning: discard trees with $p < 1/n$

	NP 0:1	NP 0:2	VP 1:2	...
Tree 1	1	0	0	...
Tree 2	1	1	0	...
Tree 3	0	1	1	...



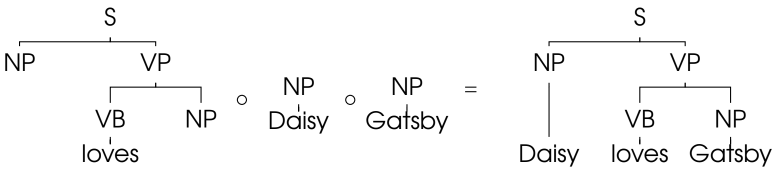
Osborne & Baldridge (EMNLP 2004),
Ensemble-based Active Learning for Parse Selection

User interface



- parser obtains n-best trees
- user walks through decision tree
  or: edit tree manually
- user accepts tree;
  grammar is augmented with fragments of this tree
  before parsing next sentence

Why DOP



- Memory-based, "training" is conceptually simple & cheap:
  new tree ⇒ extract fragments ⇒ update grammar
- Incremental model fitting more challenging/expensive with other methods:
  - Split-merge grammars (EM),
  - Bayesian grammars (Gibbs sampling),
  - Deep Learning (SGD).

Bod (1992); Sangati & Zuidema (EMNLP 2011): 2DOP

Augmenting the grammar

Given a new tree T and the current grammar G, a multiset of tree fragments.

- extract recurring fragments among initial training set and new tree
- new fragment compile into new, unique rules
  existing fragment increment relative frequency of existing rules
- bookkeeping: re-normalize grammar, re-sort indexes of rules, etc.

Typically takes < 1 second to add 1 parse tree to the grammar.

van Cranenburgh (2014): (tree) fragments (in) linear avg. time

Robustness

How to avoid dreaded "no parse"?

- Ideally, a statistical parser finds a parse tree for any input
- However, when grammar contains discontinuous constituents, function tags, not all productions may be available.

Workaround: extract partial parses from incomplete chart w/recursive algorithm:

1. Extract largest, most probable subtree from chart
2. Repeat for rest of sentence

Results become siblings under ROOT label.

Pilot experiment

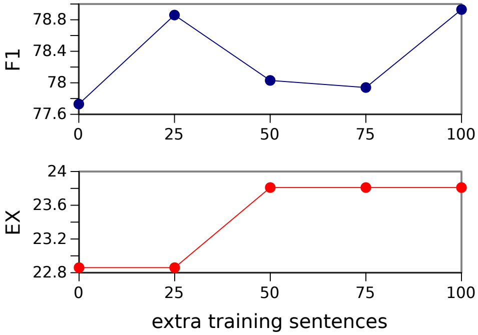
- initial grammar: DOP grammar of FTB
  (13k sentences *Le Monde* newspaper)

	F1	POS %
2DOP, Sangati & van Cra. (2015)	79.3	96.3
Stanford parser, Green et al (2013)	79.0	

- new data: first 2 chapters of *Madame Bovary*
  (Flaubert 1856, 215 sentences).
  Annotated by yours truly.
- 50% split of new trees:
  extra train trees, test set

Evaluation

Model, train set	Test set	F1	EX
2DOP, FTB	FTB	79.3	19.9
2DOP, FTB	Bovary	77.7	22.9
2DOP, FTB + 100 Bovary trees	Bovary	**78.9**	**23.8**



- out-of-domain effect is small: 7 % rel. error increase
- 5 % relative error reduction from just 100 new trees

Observations about annotation / UI

- Decision tree useful to guide attention,
  but for obvious mistakes, editing is faster.
- Long sentences don't fit on screen ...
- Partial parses not very good.
- Inconsistent parses, e.g. multiple subjects.

Sketch of larger experiment

- Grimm's fairy tales (how many sentences?)
- Multiple annotators (how many?)
- Measure:
  - effect of order of annotation:
    original, random, ranked
  - Track time/mouse clicks per sentence

Conclusion

Yes, we can ...

Conclusion

Yes, we can ...

Make Annotation Great Again!

- Encouraging results:
  - Literary, out-of-domain text parsed relatively well
  - Small number of annotations already improve accuracy
- More comprehensive experiments needed to see to what extent incremental learning really helps

Code will be made available at
`http://github.com/andreasvc/disco-dop`

Possible improvements

General:

- Gamification: encourage inter-annotator agreement
- Optimize workflow; keyboard-based UI

Ideas from previous work:

- Osborne & Baldridge (EMNLP 2004):
  - Use diverse ensemble of parsers
- Baldridge & Palmer (EMNLP 2009):
  - Model annotator expertise/fallibility
  - Model cost of annotation given sentence
- Mirroshandel & Nasr (IWPT 2011):
  - Rank per-token uncertainty instead of by sentence

Wild ideas

- Bootstrap a new treebank when no initial grammar is available? (endangered / low-resource languages)
- Add new levels of annotation to an existing treebank? e.g.,
  - multi-word expressions
  - semantic frames etc.
- Joint annotation of constituency and dependency structures?
- Grammar engineering instead of treebank annotation; e.g., LTAG, RRG