Corpus-Driven PCFG Approaximation of ERG
50 Million Trees, Is It Enough?

Yi Zhang

LT-Lab, DFKI, Germany

Computational Linguistics, Saarland University, Germany

Motivation

- Robustness: Current ERG is typically missing 10∼20% raw coverage
- Specificity: The accuracy of the HPSG parser is further offset by the accuracy of the disambiguation model (another ∼20% of trees are rejected)
- Efficiency: No polynomial time complexity upper-bound in unification-based parsing. Practically a parser can be never too fast

A CFG Approximation

- Complex structures in a deeper formalism (TFS) can be reduced to non-recursive/atomic categories
- Use context-free rewriting rules to capture the constructions
- Previous attempts
  - Grammar-driven approach: compiling out CFG rules directly from the HPSG grammar [Kiefer and Krieger, 2004]
  - Corpus-driven approach: acquiring CFG from corpus observations [Krieger, 2007]
  - Main problems: huge CFG makes processing intractable

A CFG Approximation

- Complex structures in a deeper formalism (TFS) can be reduced to non-recursive/atomic categories
- Use context-free rewriting rules to capture the constructions
- Previous attempts
  - Grammar-driven approach: compiling out CFG rules directly from the HPSG grammar [Kiefer and Krieger, 2004]
  - Corpus-driven approach: acquiring CFG from corpus observations [Krieger, 2007]
  - Main problems: huge CFG makes processing intractable

* Compare the SF restrictor and ambiguity packing mechanisms

Corpus-Driven (P)CFG Approximation

- Approximate both the HPSG grammar and the disambiguation model with a single (generative) probabilistic CFG
- Only use one preferred reading per sentence (either the gold tree from manual disambiguation, or the top-ranked reading by the MaxEnt)

(P)CFG Extraction from Treebank

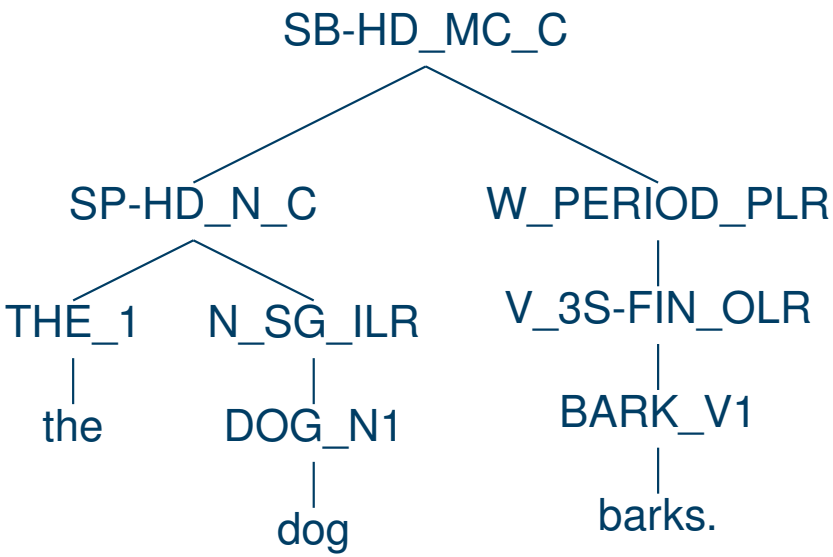
- A derivation tree records a complete analysis (except for the CM preprocessing information)
- CFG categories and rules are extracted from “decorated” derivation trees
- Estimation of probabilities: Naïve Maximal Likelihood Estimate, a.k.a. counting the rules
  - Constructions: no smoothing
  - Lexicon: $P(w|t)$ is smoothed to take care of unknown words
    No context is used. No sequence tagging

“Decorations”

- Internal “decorations”: information from feature structures [Krieger, 2007]
- External “decorations”: information from ancestors, off-springs, siblings, etc. [Klein and Manning, 2003]

External “Decorations”

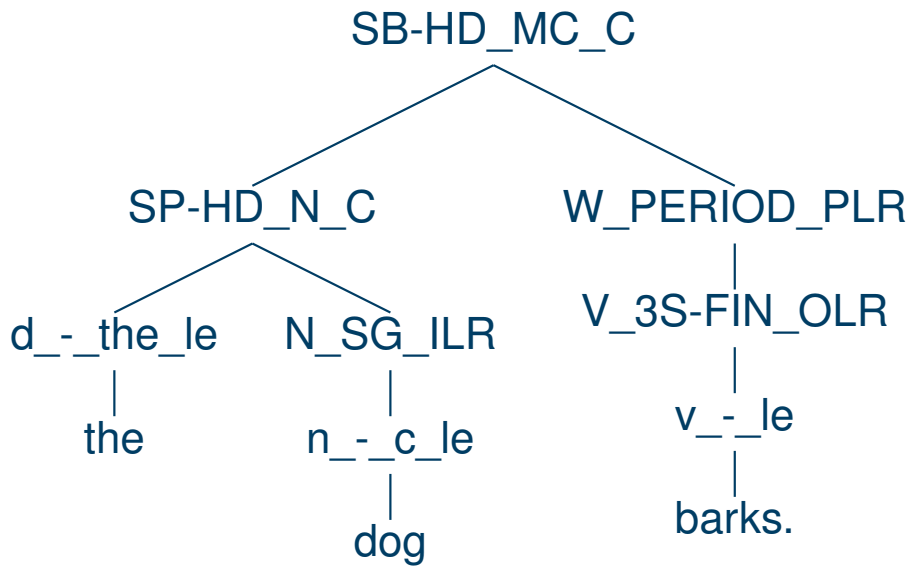
1. Replace Lexical entry names with corresponding LE type
2. Collapse morphological rules with the LE type to form “supertags”
3. Split punctuation into separate nodes in the derivation tree
4. Grand-parenting (vertical Markovization)



DELPH-IN

External “Decorations”

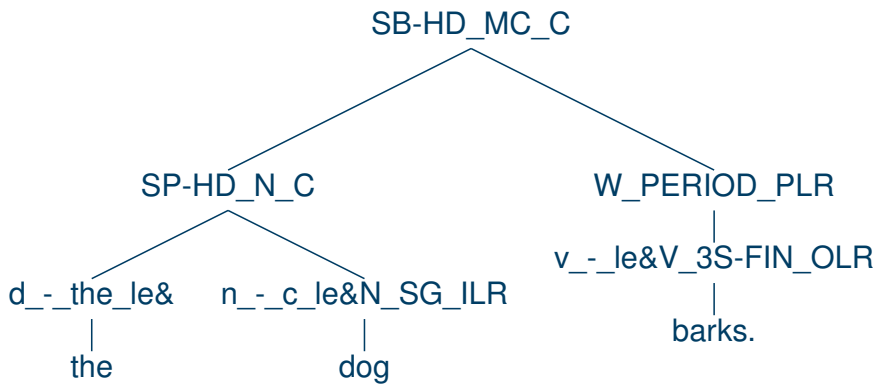
1. Replace Lexical entry names with corresponding LE type
2. Collapse morphological rules with the LE type to form “supertags”
3. Split punctuation into separate nodes in the derivation tree
4. Grand-parenting (vertical Markovization)



DELPH-IN

External “Decorations”

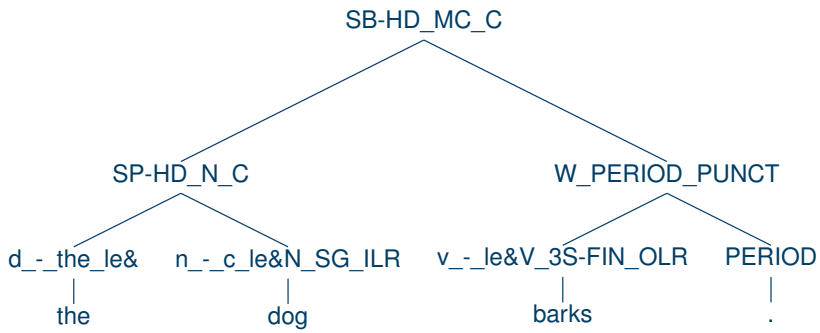
1. Replace Lexical entry names with corresponding LE type
2. Collapse morphological rules with the LE type to form “supertags”
3. Split punctuation into separate nodes in the derivation tree
4. Grand-parenting (vertical Markovization)



DELPH-IN

External “Decorations”

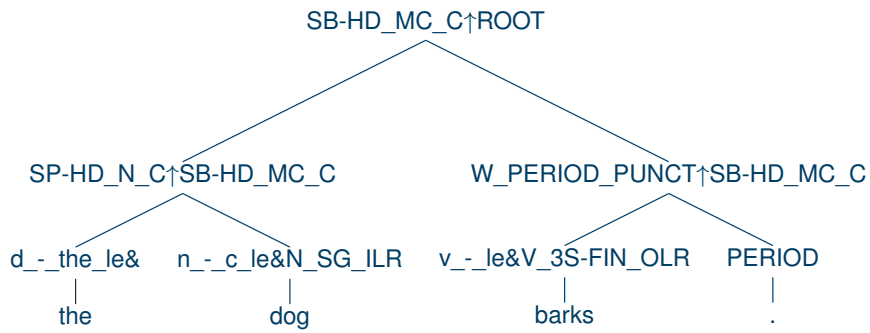
1. Replace Lexical entry names with corresponding LE type
2. Collapse morphological rules with the LE type to form “supertags”
3. Split punctuation into separate nodes in the derivation tree
4. Grand-parenting (vertical Markovization)



DELPH-IN

External “Decorations”

1. Replace Lexical entry names with corresponding LE type
2. Collapse morphological rules with the LE type to form “supertags”
3. Split punctuation into separate nodes in the derivation tree
4. Grand-parenting (vertical Markovization)



DELPH-IN

The Parser

Jigsaw

- A BitPar Java implementation, using bit-vector operations for efficient “parallel” recognition
- The best reading is recovered with Viterbi decoding
- Works with millions of CFG rules and hundred thousands of NTs without pruning
- Supports word lattice input
- Training PCFG on WSJ takes 10 seconds
- Reproduces almost exactly the same results as reported by [Klein and Manning, 2003] on WSJ parsing

Experiment

- ERG (1010)
- Train approximating PCFG with LOGON/WeScience
- Train approximating PCFG with WikiWoods (top-1 reading selected by the MaxEnt model)
- ParsEval labeled bracketing F1, exact match rate and tagging accuracy are used as quality measures

Evaluation

	#P	#NT	#T	P	R	F1	EM	TA
LOGON-nop	14963	1409	1782	71.17	60.68	65.51	23.96	78.01
LOGON	15965	2597	987	73.26	63.16	67.84	24.85	83.18
WS+LOGON	21899	2193	1152	75.99	71.94	73.91	28.40	87.16
				72.66	66.00	69.17	16.91	84.90
WW0	479850	7570	3464	79.69	79.04	79.36	27.64	93.24
WW	1007563	8852	4472	80.34	79.60	79.97	28.79	93.45

- LOGON: trained with LOGON (7027 trees)
- -nop: without treatment of punctuations
- WS: trained with WS (7636 trees)
- WW0: trained with 10% of WW (∼4.8M trees)
- WW: trained with the entire WW (∼48M trees)

Conclusion

- The syntactic constructions are still coarse-grained with only 1 level of GP
- Supertagset is over-finegrained when using LOGON+WS for training, but tagging accuracy is acceptable with WW
- Preliminary experiments of adding constituent categories to syntactic constructions shows improvement (2% EM)
- Approximating PCFG continues to grow after 50M trees, but most new rules involve combination of complex supertags and have few impact on parsing accuracy

aj-hdn_norm_c↑sp-hd_n_c → v_np-pp*_to_le&v_pas_odlr&v_v-un_dlr&v_j-nb-pas-tr_dlr@
n_-_c-pl-nocnh_le&n_pl_olr@

- Comparison with Berkeley parser shows that our MLE PCFG trained with 50M trees is superior

Future Work

- Try GP level 2
- Evaluation with EDM
- Play with CFG pruning?
- Rethink about the current MaxEnt model which uses surprising few non-local features (in comparison with Charniak&Johnson Reranking features)
- Comparison with internal “decorations”
- Integration with standard DELPH-IN tools

References I

Kiefer, B. and Krieger, H.-U. (2004).
A context-free superset approximation of unification-based grammars.
In *New Developments in Parsing Technology*. Kluwer.
http://www.wkap.nl/prod/b/1-4020-2293-X.

Klein, D. and Manning, C. D. (2003).
Accurate unlexicalized parsing.
In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 423–430, Sapporo, Japan.

Krieger, H.-U. (2007).
From ubgs to cfgs a practical corpus-driven approach.
Natural Language Engineering, 13(4):317–351.
Published online in April 2006.