Statistical Parsing

Paper presentation:

Michael Collins (2003). "Head-driven statistical models for natural language parsing". In: *Computational linguistics* 29.4, pp. 589–637. DOI: 10.1162/089120103322753356

Çağrı Çöltekin

University of Tübingen
Seminar für Sprachwissenschaft

December 2016

What is the paper about?

- A head-driven, lexicalized PCFG
- PCFGs cannot capture many linguistic phenomena
- Lexicalizing PCFGs allows capturing lexical dependencies, but parameter estimation becomes difficult (many rules, sparse data)
- The main idea is factoring the rule probabilities, into parts that are easy to estimate
- The paper does that in a linguistically-motivated way
- The resulting parser works better than PCFGs, and some others in the literature

Three models

Model 1
- Lexicalize the PCFG
- Condition the probability of a rule based on parts of its LHS
- Condition probabilities of non-heads on distance to their head

Model 2 Add complement-adjunct distinction (use subcategorization frames)

Model 3 Add conditions for wh-movement

An overview of the paper

2. Background: PCFGs, lexicalization, estimation (MLE)
3. Model definitions
4. Special cases: mainly related to treebank format
5. Practical issues: parameter estimation, unknown words, parsing algorithm
6. Results
7. Discussion
8. Related work
9. Conclusions

Probabilistic context-free grammars

- A CFG augmented with probabilities for each rule
- Assigns a proper probability distribution to parse trees
  - if all rule probabilities with the same LHS sum to 1
  - all derivations terminate in a finite number of steps
- The main problem is estimating probabilities associated with each rule $X \rightarrow \beta$
- Maximum-likelihood estimate:

$$\frac{\text{count}(X \rightarrow \beta}{\text{count}(X)}$$

- With rule probabilities, parsing is finding the best tree

$$T_{\text{best}} = \arg\max_{T} P(T|S) = \arg\max_{T} \frac{P(T,S)}{P(S)} = \arg\max_{T} P(T,S)$$

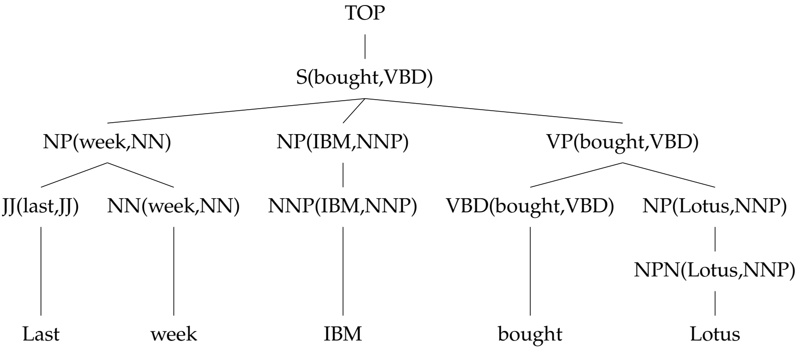
Probabilistic context-free grammars (2)

- In PCFGs derivations are assumed to be independent
- The probability of a tree is the product of the probabilities of rules used in the derivation
- PCFGs cannot capture lexical or structural dependencies

Lexicalizing PCFGs

- Replace non-terminal X with $X(h)$, where h is a tuple with the lexical word and its POS tag
- Now the grammar can capture (head-driven) lexical dependencies
- But number of nonterminals grow by $|V| \times |T|$
- Estimation becomes difficult (many rules, data sparsity)
- Note: Penn Treebank (PTB) does not annotate heads, they are automatically annotated (based on heuristics)

Example lexicalized derivation



Example rules:

TOP	$\rightarrow$	S(bought,VBD)
S(bought,VBD)	$\rightarrow$	NP(week,NN) NP(IBM,NNP) VP(bought,VBD)
VP(bought,VBD)	$\rightarrow$	VBD(bought,VBD) NP(Lotus,NNP)
JJ(last,JJ)	$\rightarrow$	Last

Model 1: the generative story

We take each lexicalized CF rule is formed as

$$X(h) \rightarrow \langle\text{left-dependents}\rangle\ H(h)\ \langle\text{right-dependents}\rangle$$

1. Generate the head with probability $P_h(H|X, h)$
2. Generate the left modifier(s) independently, each with probability $P_l(L_i(l_i)|X, h, H)$
3. Generate the left modifier(s) independently, each with probability $P_r(R_i(r_i)|X, h, H)$

- A special left/right dependent label 'STOP' terminates the generation

Model 1: distance

- Model 1, also conditions the left and right dependents on their distance from the head. For example P_l is estimated using
$$P_l(L_i(l_i)|X, h, H, \mathit{distance}(i-1))$$
- Two distance measures:
  - Is the intervening string length 0? (adjacency)
  - Does the intervening string contain a verb? (clausal modifiers)

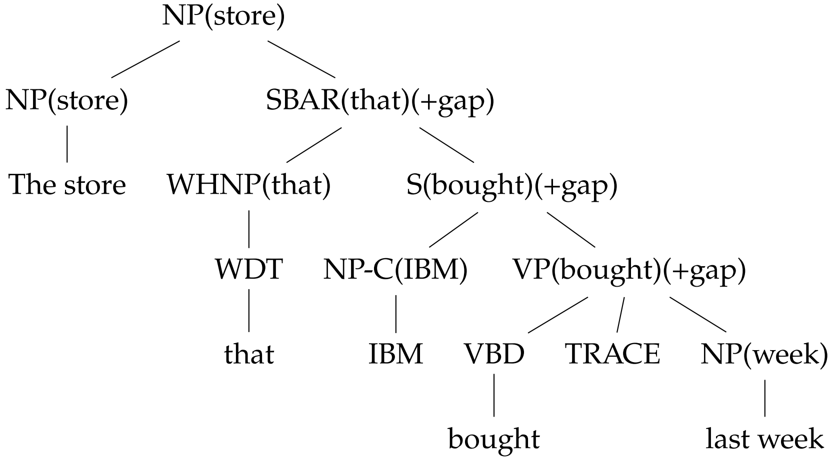
Model 2: the generative story

Main idea: condition the right/left modifiers on subcategorization frames (LC and RC), which are the left and right complements of the head.

1. Generate the head with probability $P_h(H|X, h)$
2. Choose left and aright subcategorization frames, with probabilities $P_{lc}(LC|X, H, h)$ and $P_{rc}(RC|X, H, h)$
3. Generate the left/right modifier(s) independently, each with probability $P_l(L_i(l_i)|X, h, H, LC)$ and $P_r(R_i(R_i)|X, h, H, RC)$

Model 3: traces and wh-movement

The idea: mark and propagate ‘gaps’.



Special cases

- Non-recursive (base) NPs are marked as NPB
- Coordination: allow only a single phrase after a CC
- Punctuation: remove all except non-initial/non-final comma and colon, treat the rest as coordination
- Empty subjects: introduce a dummy empty subject during preprocessing

Parameter estimation

Parameters are estimated by three levels of backoff (see Table 1 in the paper for details), using a version of Witten-Bell smoothing

$$e = \lambda_1 e_1 + (1 - \lambda_1)(\lambda_2 e_2 + (1 - \lambda_2) e_3)$$

where,

$$\lambda_1 = \frac{f_1}{f_1 + 5u_1}$$

f_1 is the relevant number of tokens (count in denominator), u_1 is the relevant number of types.
Other λs are calculated similarly.

Unknown words and parsing algorithm

- During training, all words with frequencies less than 6 were replaced with `UNKNOWN`
- During testing, the POS tags for unknown words were assigned using using the tagger by Ratnaparkhi (1996)
- The parsing algorithm is a version of CKY parser with $O(n^5 0$ complexity

Results

- Model 2 performs better than Model 1
- Model 2 also performs better/similar in comparison to earlier/state-of-the-art models
- Details: Table 2 on page 608 on paper.

More on results

- Phrase-label precision/recall results do not show attachment problems.
- Extracted dependencies are more useful (Figure 12 on page 610)
- The parser recovers 'core' dependencies successfully,
- Main problems are with adjuncts and coordination

More on distance measure

- Distance measure seem to help finding subcategorization for Model 1
- As the distance from the head increases,
  - the probability of attaching a new modifier decreases
  - the probability of attaching 'STOP' increases
- Distance measure is also useful for preferring right-branching
- Structural (e.g., close attachment) vs. lexical/semantic preferences: structural preferences seem to be necessary. For example:

  `John was believed to have been shot by Bill`

  `Flip said that Squeaky will do the work yesterday`

Choice of representation

- The parser prefers PTB-style (flat) trees
- For binary representations, do pre-/post-processing
- This would have an effect on capturing structural (but not lexical) preferences.
- Preprocessing steps, e.g., `NPB` labeling, seem to be important
- In general, the parser works best with
  - flat trees
  - different constituent labels at different levels

The need to break down rules

- The main benefit is the parser can use rules that it has not seen in the training data
- The parser can also learn some regularities in the rules
- Compare with Charniak (1997) which only allows rules seen in the training data
- This is more important for PTB,

PTB	**alternative**
$VP \rightarrow V\ NP$	$VP \rightarrow V\ NP$
$VP \rightarrow V\ NP\ PP$	$VP \rightarrow VP\ PP$
$VP \rightarrow V\ NP\ PP\ PP$	
$\ldots$	

- In PTB, 54.5% of the rules (of the form used by this parser) only occur once

Summary

- Accurate generative parser that breaks down rules
- Does well on ‘core’ dependencies, adjuncts and coordination are the main sources of error
- Either conditioning on adjacency or subcategorization is needed for good accuracy
- The models work well with flat dependencies
- Breaking down the rules have good properties (can use rules that were not seem in the training)

Bibliography

Collins, Michael (2003). “Head-driven statistical models for natural language parsing”. In: *Computational linguistics* 29.4, pp. 589–637. DOI: 10.1162/089120103322753356.

Ratnaparkhi, Adwait (1996). “A maximum entropy model for part-of-speech tagging”. In: *Proceedings of the conference on empirical methods in natural language processing*. Vol. 1, pp. 133–142.