Probabilistic Parsing

FSNLP, chapter 12

Christopher Manning and Hinrich Schütze

Modern Statistical Parsers

- A greatly increased ability to do accurate, robust, broad coverage parsing (Charniak 1997; Collins 1997; Ratnaparkhi 1997b; Charniak 2000)
- Achieved by converting parsing into a classification task and using statistical/machine learning methods
- Statistical methods (fairly) accurately resolve structural and real world ambiguities
- Much faster: rather than being cubic in the sentence length or worse, for modern statistical parsers parsing time is made linear (by using beam search)
- Provide probabilistic language models that can be integrated with speech recognition systems.

Supervised ML parsing

- Crucial resource has been treebanks of parses, especially the Penn Treebank (Marcus et al. 1993)
- From these train classifiers:
  - Mainly probabilistic models, but also:
  - Conventional decision trees
  - Decision lists/transformation-based learning
- Possible only when extensive resources exist
- Somewhat uninteresting from Cog. Sci. viewpoint – which would prefer bootstrapping from minimal supervision

A Penn Treebank tree (POS tags not shown)

```
( (S (NP-SBJ The move)
     (VP followed
         (NP (NP a round)
             (PP of
                 (NP (NP similar increases)
                     (PP by
                         (NP other lenders))
                     (PP against
                         (NP Arizona real estate loans)))))
         ,
         (S-ADV (NP-SBJ *)
                (VP reflecting
                    (NP (NP a continuing decline)
                        (PP-LOC in
                                (NP that market))))))
     .))
```


Probabilistic models for parsing

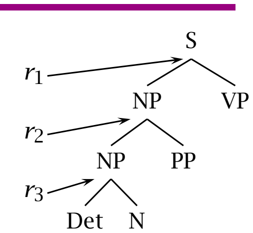
- **Conditional/Parsing model:** We estimate directly the probability of parses of a sentence

  $\hat{t} = \arg\max_t P(t|s,G)$ where $\sum_t P(t|s,G) = 1$
- We don't learn from the distribution of sentences we see (but nor do we assume some distribution for them)
  - (Magerman 1995; Collins 1996; **?**)
- **Generative/Joint/Language model:**

  $$\sum_{\{t:\ \mathrm{yield}(t)\in\mathcal{L}\}} P(t) = 1$$
- Most likely tree

  $\hat{t} = \arg\max_t P(t|s) = \arg\max_t \frac{P(t,s)}{P(s)} = \arg\max_t P(t,s)$
  - (Collins 1997; Charniak 1997, 2000)

Generative/Derivational model = Chain rule



$$P(t) = \sum_{\{d:\ d \text{ is a derivation of } t\}} P(d)$$

Or: $P(t) = P(d)$ where d is the canonical derivation of t

$$d = P(\mathrm{S} \xrightarrow{r_1} \alpha_1 \xrightarrow{r_2} \ldots \xrightarrow{r_m} \alpha_m = s) = \prod_{i=1}^{m} P(r_i|r_1,\ldots r_{i-1})$$

- History-based grammars

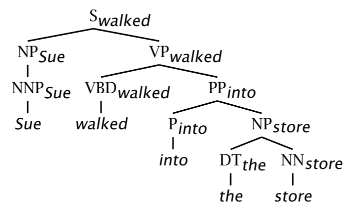
  $$P(d) = \prod_{i=1}^{m} P(r_i|\pi(h_i))$$

Enriching a PCFG

- A naive PCFG with traditional nonterminals (NP, PP, etc.) works quite poorly due to the independence assumptions it embodies (Charniak 1996)
- Fix: encode more information into the nonterminal space
  - Structure sensitivity (Manning and Carpenter 1997; Johnson 1998b)
    - Expansion of nodes depends a lot on their position in the tree (independent of lexical content)
    - E.g., enrich nodes by also recording their parents: ^{S}NP is different to ^{VP}NP

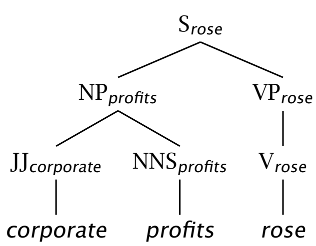
Enriching a PCFG (2)

- (Head) Lexicalization (Collins 1997; Charniak 1997)
  - The head word of a phrase gives a good representation of the phrase's structure and meaning
  - Puts the properties of words back into a PCFG

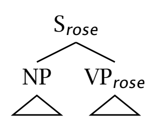


Parsing via classification decisions: Charniak (1997)

- A very simple, conservative model of lexicalized PCFG
- Probabilistic conditioning is "top-down" (but actual computation is bottom-up)



Charniak (1997) example

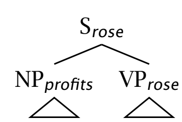


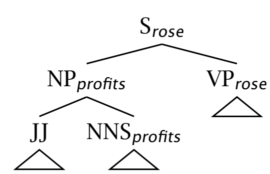
a. h = *profits*; c = NP

b. ph = *rose*; pc = S

c. $P(h|ph, c, pc)$

d. $P(r|h, c, pc)$





Charniak (1997) linear interpolation/shrinkage

$$\begin{aligned}\hat{P}(h|ph,c,pc) &= \lambda_1(e)P_{\mathsf{MLE}}(h|ph,c,pc)\\ &\quad+\lambda_2(e)P_{\mathsf{MLE}}(h|C(ph),c,pc)\\ &\quad+\lambda_3(e)P_{\mathsf{MLE}}(h|c,pc)+\lambda_4(e)P_{\mathsf{MLE}}(h|c)\end{aligned}$$

- $\lambda_i(e)$ is here a function of how much one would expect to see a certain occurrence, given the amount of training data, word counts, etc.
- $C(ph)$ is semantic class of parent headword
- Techniques like these for dealing with data sparseness are vital to successful model construction

Charniak (1997) shrinkage example

	$P(\text{prft}\mid\text{rose}, \text{NP}, \text{S})$	$P(\text{corp}\mid\text{prft}, \text{JJ}, \text{NP})$
$P(h\mid ph, c, pc)$	0	0.245
$P(h\mid C(ph), c, pc)$	0.00352	0.0150
$P(h\mid c, pc)$	0.000627	0.00533
$P(h\mid c)$	0.000557	0.00418

- Allows utilization of rich highly conditioned estimates, but smoothes when sufficient data is unavailable
- One can't just use MLEs: one commonly sees previously unseen events, which would have probability 0.

Sparseness & the Penn Treebank

- The Penn Treebank – 1 million words of parsed English *WSJ* – has been a key resource (because of the widespread reliance on supervised learning)
- But 1 million words is like nothing:
  - 965,000 constituents, but only 66 WHADJP, of which only 6 aren't *how much* or *how many*, but there is an infinite space of these (*how clever/original/incompetent* (*at risk assessment and evaluation*))
- Most of the probabilities that you would like to compute, you can't compute

Sparseness & the Penn Treebank (2)

- Most intelligent processing depends on bilexical statistics: likelihoods of relationships between pairs of words.
- Extremely sparse, even on topics central to the *WSJ*:
  - stocks plummeted 2 occurrences
  - stocks stabilized 1 occurrence
  - stocks skyrocketed 0 occurrences
  - #stocks discussed 0 occurrences
- So far there has been very modest success augmenting the Penn Treebank with extra unannotated materials or using semantic classes or clusters (cf. Charniak 1997, Charniak 2000) – as soon as there are more than tiny amounts of annotated training data.

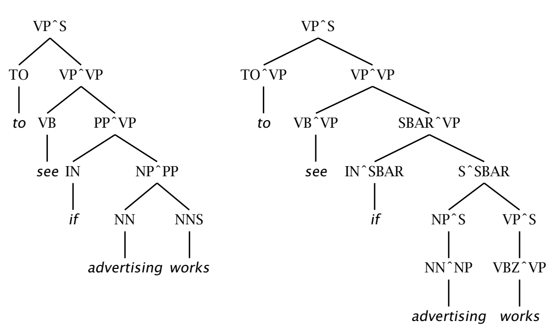
Probabilistic parsing

- Charniak (1997) expands each phrase structure tree in a single step.
- This is good for capturing dependencies between child nodes
- But it is bad because of data sparseness
- A pure dependency, one child at a time, model is worse
- But one can do better by in between models, such as generating the children as a Markov process on both sides of the head (Collins 1997; Charniak 2000)

Correcting wrong context-freedom assumptions

		Horizonal Markov Order				
Vertical Order		$h = 0$	$h = 1$	$h \le 2$	$h = 2$	$h = \infty$
$v = 0$	No annotation	71.27 (854)	72.5 (3119)	73.46 (3863)	72.96 (6207)	72.62 (9657)
$v \le 1$	Sel. Parents	74.75 (2285)	77.42 (6564)	77.77 (7619)	77.50 (11398)	76.91 (14247)
$v = 1$	All Parents	74.68 (2984)	77.42 (7312)	77.81 (8367)	77.50 (12132)	76.81 (14666)
$v \le 2$	Sel. GParents	76.50 (4943)	78.59 (12374)	79.07 (13627)	78.97 (19545)	78.54 (20123)
$v = 2$	All GParents	76.74 (7797)	79.18 (15740)	79.74 (16994)	79.07 (22886)	78.72 (22002)

Correcting wrong context-freedom assumptions



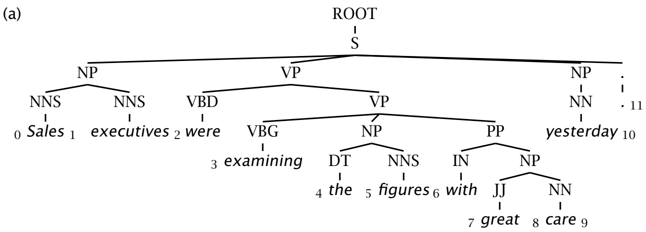
(a) (b)

Correcting wrong context-freedom assumptions

	Cumulative			Indiv.
Annotation	Size	F_1	ΔF_1	ΔF_1
Baseline	7619	77.72	0.00	0.00
UNARY-INTERNAL	8065	78.15	0.43	0.43
UNARY-DT	8078	80.09	2.37	0.22
UNARY-RB	8081	80.25	2.53	0.48
TAG-PA	8520	80.62	2.90	2.57
SPLIT-IN	8541	81.19	3.47	2.17
SPLIT-AUX	9034	81.66	3.94	0.62
SPLIT-CC	9190	81.69	3.97	0.17
SPLIT-%	9255	81.81	4.09	0.20
TMP-NP	9594	82.25	4.53	1.12
GAPPED-S	9741	82.28	4.56	0.22
POSS-NP	9820	83.06	5.34	0.33
SPLIT-VP	10499	85.72	8.00	1.41
BASE-NP	11660	86.04	8.32	0.78
DOMINATES-V	14097	86.91	9.19	1.47
RIGHT-REC-NP	15276	87.04	9.32	1.99

Evaluation

(a)



(b) Brackets in gold standard tree (a.):
S-(0:11), **NP-(0:2)**, VP-(2:9), VP-(3:9), **NP-(4:6)**, PP-(6-9), NP-(7,9), *NP-(9:10)

(c) Brackets in candidate parse:
S-(0:11), **NP-(0:2)**, VP-(2:10), VP-(3:10), NP-(4:10), **NP-(4:6)**, PP-(6-10), NP-(7,10)

(d)

Precision:	3/8 = 37.5%	Crossing Brackets:	0
Recall:	3/8 = 37.5%	Crossing Accuracy:	100%
Labeled Precision:	3/8 = 37.5%	Tagging Accuracy:	10/11 = 90.9%
Labeled Recall:	3/8 = 37.5%		

Parser results

- Parsers are normally evaluated on the relation between *individual postulated nodes* and ones in the gold standard tree (Penn Treebank, section 23)
- Normally people make systems balanced for precision/recall
- Normally evaluate on sentences of 40 words or less
- Magerman (1995): about 85% labeled precision and recall
- Charniak (2000) gets 90.1% labeled precision and recall
- Good performance. Steady progress in error reduction
- At some point size of and errors in treebank must become the limiting factor
  - (Some thought that was in 1997, when several systems were getting 87.x%, but apparently not.)