Probabilistic Parsing

FSNLP, chapter 12

Christopher Manning and Hinrich Schütze
© 1999–2002

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

Probabilistic models for parsing

- Conditional/Parsing model: We estimate directly the probability of parses of a sentence
  \[
  \hat{t} = \text{arg max}_t P(t|s, G) \quad \text{where} \quad \sum_t P(t|s, G) = 1
  \]
- 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:
  \[
  \sum_{t: \text{yield}(t) \in \mathcal{L}} P(t) = 1
  \]
- Most likely tree
  \[
  \hat{t} = \text{arg max}_t P(t|s) = \text{arg max}_t \frac{P(t,s)}{P(s)} = \text{arg max}_t P(t,s)
  \]
  - (Collins 1997; Charniak 1997, 2000)

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))))))

Generative/Derivational model = Chain rule

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

Or:

\[
P(t) = P(d) \quad \text{where } d \text{ is the canonical derivation of } t
\]

\[
d = P(S \mid \alpha_1 \alpha_2 \ldots \alpha_m = s) = \prod_{i=1}^{m} P(r_i | r_1, \ldots r_{i-1})
\]

- 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 $V_PNP$

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) linear interpolation/shrinkage

$P(h|ph,c,pc) = \lambda_1(e)P_{MLE}(h|ph,c,pc) + \lambda_2(e)P_{MLE}(h|C(ph),c,pc) + \lambda_3(e)P_{MLE}(h|c,pc) + \lambda_4(e)P_{MLE}(h|c)$

- $\lambda_i(e)$ 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

- 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

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

• 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.
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.)