ACS Introduction to NLP
Lecture 4a: Statistical Parsing I

Stephen Clark

Natural Language and Information Processing Research Group

sc609@cam.ac.uk

Interesting Ambiguity Examples

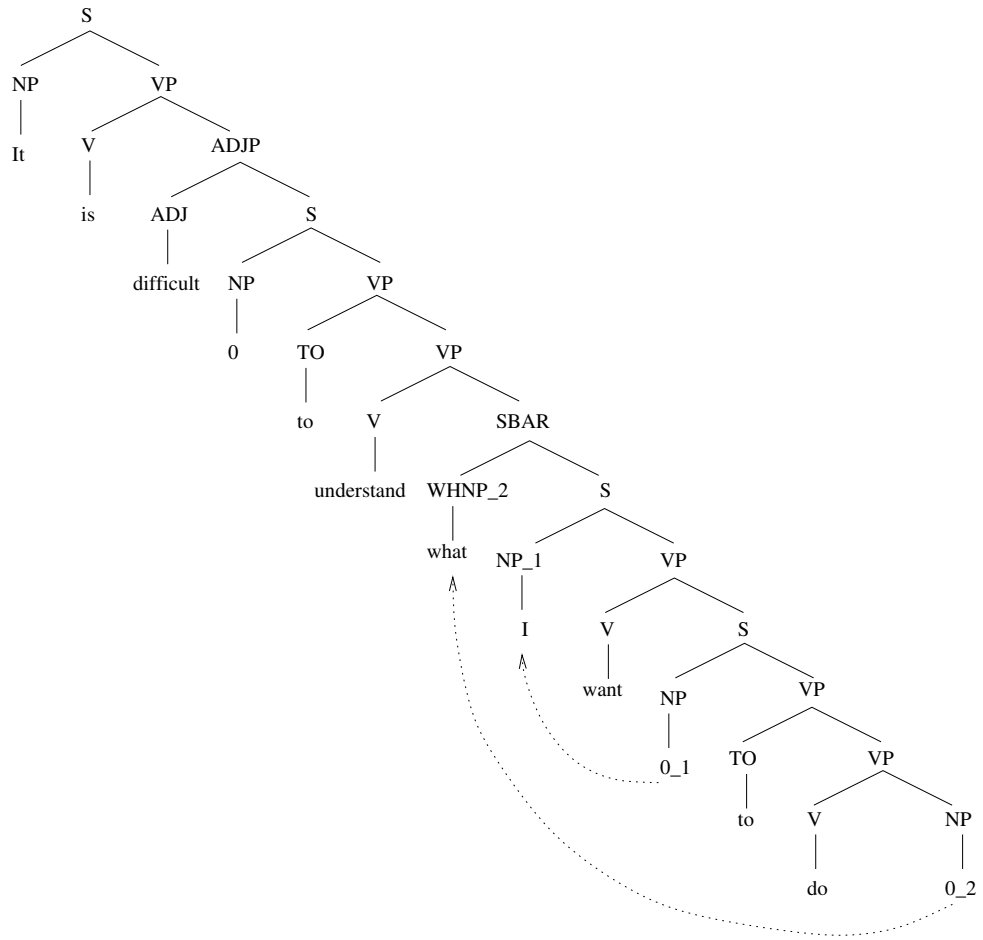
- The a are of I
- The cows are grazing in the meadow
- John saw Mary

examples from Abney (1996)

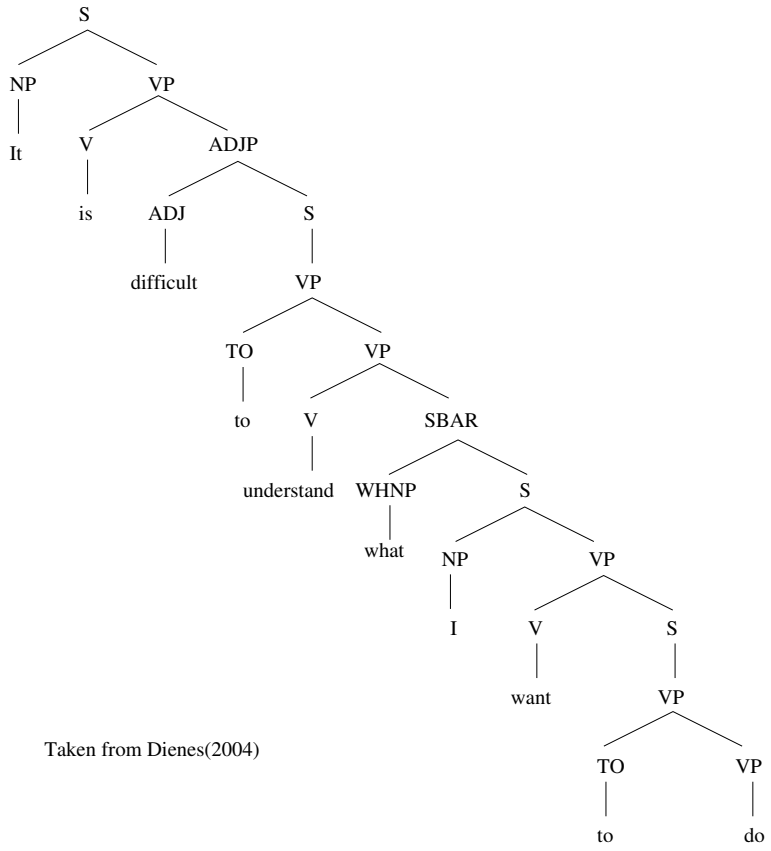
The Penn Treebank

- 40,000 sentences of WSJ newspaper text annotated with phrase-structure trees
- The trees contain some predicate-argument information and traces
- Created in the early 90s
- Produced by automatically parsing the newspaper sentences followed by manual correction
- Took around 3 years to create
- Sparked a parsing "competition" which is still running today
  - leading some commentators to describe the last 10 years of NLP as the study of the WSJ

An Example Penn Treebank Tree



A Tree a typical PTB Parser would produce



Taken from Dienes(2004)

Characterisation of Statistical Parsing 1

- What is the grammar which determines the set of legal syntactic structures for a sentence? How is that grammar obtained?
- What is the algorithm for determining the set of legal parses for a sentence (given a grammar)?
- What is the model for determining the plausibility of different parses for a sentence?
- What is the algorithm, given the model and a set of possible parses, which finds the best parse?

Characterisation of Statistical Parsing 2

$$T_{\text{best}} = \arg\max_{T} \mathit{Score}(T, S)$$

- Just two components:
  - the *model*: a function *Score* which assigns scores (probabilities) to tree, sentence pairs
  - the *parser*: the algorithm which implements the search for T_{best}
- Statistical parsing seen as more of a pattern recognition/Machine Learning problem plus search
  - the grammar is only implicitly defined by the training data and the method used by the parser for generating hypotheses

Statistical Parsing Models

- Probabilistic approach would suggest the following for the *Score* function:

$$\mathit{Score}(T, S) = P(T|S)$$

- Lots of research on different probability models for Penn Treebank trees
  - generative models, log-linear (maximum entropy) models, perceptron, ...

Generative Models

$$\begin{aligned} \arg\max_T P(T|S) &= \arg\max_T \frac{P(T,S)}{P(S)} \\ &= \arg\max_T P(T,S) \end{aligned}$$

- Why model the joint probability when the sentence is given?
- Modelling a parse as a generative process allows the parse to be broken into manageable parts, for which the corresponding probabilities can be reliably estimated
- Probability estimation is easy for these sorts of models (ignoring smoothing issues)
  - maximum likelihood estimation = relative frequency estimation
- But choosing how to break up the parse is something of a black art

PCFGs

- Probabilistic Context Free Grammars provide a ready-made solution to the statistical parsing problem
- However, it is important to realise that **parameters do not have to be associated with the rules of a context free grammar**
  - we can choose to break up the tree in any way we like
- But extracting a PCFG from the Penn Treebank and parsing with it provides a useful baseline
  - a PCFG parser obtains roughly 70-75% Parseval scores

Parameterisation of a Parse Tree

- Collins describes the following two criteria for a good parameterisation:
  - **Discriminative power**: the parameters should include the contextual information required for the disambiguation process
    (PCFGs fail in this regard)
  - **Compactness**: the model should have as few parameters as possible (while still retaining adequate discriminative power)

- Collins thesis (p.11) has an instructive belief networks example, which shows that choice of variable ordering is crucial

Generative Models for Parse Trees

- **Representation**
  - the set of part-of-speech tags
  - whether to pass lexical heads up the tree (lexicalisation)
  - whether to replace words with their morphological stems
- **Decomposition**
  - the order in which to generate the tree
  - the order of *decisions*, d_i, made in generating the tree
  - these decisions do not have to correspond to parsing decisions
- **Independence assumptions**
  - group decision sequences into equivalence classes, Φ

$$P(T, S) = \prod_{i=1}^{n} P(d_i | \Phi(d_1 \dots d_{i-1}))$$

Successive Parameterisations

- Simple PCFG
- PCFG + dependencies
- Dependencies + direction
- Dependencies + direction + relations
- Dependencies + direction + relations + subcategorisation
- Dependencies + direction + relations + subcategorisation + distance
- Dependencies + direction + relations + subcategorisation + distance + parts-of-speech

The Basic Generative Model

- Each rule in a PCFG has the following form:

$$P(h) \rightarrow L_n(l_n) \ldots L_1(l_1) H(h) R_1(r_1) \ldots R_m(r_m)$$

P is the parent; H is the head-child; L_i and R_i are left and right modifiers (n or m may be zero)

- The probability of a rule can be written (exactly) using the chain rule:

$$\begin{aligned} p(L_n(l_n) \ldots L_1(l_1) H(h) R_1(r_1) \ldots R_m(r_m) | P(h)) = \\ p(H|P(h)) \times \\ \prod_{i=1}^{n} p(L_i(l_i)|L_1(l_1) \ldots L_{i-1}(l_{i-1}), P(h), H) \times \\ \prod_{j=1}^{m} p(R_j(r_j)|L_1(l_1) \ldots L_n(l_n), R_1(r_1) \ldots R_n(r_n), P(h), H) \end{aligned}$$

Independence Assumptions

- For Model 1, assume the modifiers are generated independently of each other:

$$p_l(L_i(l_i)|L_1(l_1)\ldots L_{i-1}(l_{i-1}), P(h), H) = p_l(L_i(l_i)|P(h), H)$$

$$p_r(R_j(r_j)|L_1(l_1)\ldots L_n(l_n), R_1(r_1)\ldots R_n(r_n), P(h), H) = p_r(R_j(r_j)|P(h), H)$$

- Example rule: S(bought) $\rightarrow$ NP(week) NP(IBM) VP(bought)

p_h(VP|S,bought) $\times$ p_l(NP(IBM)|S,VP,bought) $\times$ p_l(NP(week)|S,VP,bought)
$\times$ p_l(STOP|S,VP,bought) $\times$ p_r(STOP|S,VP,bought)

Subcategorisation Parameters

- A better model would distinguish optional arguments (adjuncts) from required arguments (complements)
- In *Last week IBM bought Lotus*, *Last week* is an optional argument
- Here the verb subcategorises for an NP subject to the left and an NP object to the right
  - subjects are often omitted from subcat frames for English (because every verb has a subject in English) but we'll keep them in the model

Subcategorisation Parameters

- Probability of the rule S(bought) $\rightarrow$ NP(week) NP-C(IBM) VP(bought):

$$p_h(\text{VP}|\text{S,bought}) \times p_{lc}(\{\text{NP-C}\}|\text{S,VP,bought}) \times p_{rc}(\{\}|\text{S,VP,bought}) \times$$
$$p_l(\text{NP-C(IBM)}|\text{S,VP,bought,}\{\text{NP-C}\}) \times p_l(\text{NP(week)}|\text{S,VP,bought,}\{\}) \times$$
$$p_l(\text{STOP}|\text{S,VP,bought,}\{\}) \times p_r(\text{STOP}|\text{S,VP,bought,}\{\})$$

Results

- Model 1 achieves 87.5/87.7 LP/LR on WSJ section 23 according to the Parseval measures
- Model 2 achieves 88.1/88.3 LP/LR
- Current best scores on this task are around 91

References

- Steven Abney (1996), Statistical Methods and Linguistics, available from Abney's webpage

- Michael Collins (1999), Head-Driven Statistical Models for Natural Language Parsing