Traitements de Corpus — 5

Background Reading:

Traitements probabilistes et corpus, Traitement automatique des langues Vol. 36, 1995, no. 1–2:

Rajman (Approche probabiliste de l'analyse syntaxique), 157–199

Basic statistical phrase structure parsing
- probabilistic context-free grammar (PCFG)

Parsing using contextual probabilities
- extended PCFG, history-based grammar
- probabilistic left corner / LR parsing
- transformation-based learning of syntax

Using lexical probabilities
- stochastic lexicalised tree-adjoining grammar
- data-oriented parsing
- parsing with bi-lexical dependencies

Treebanks and Training

Treebanks vital for training in all approaches to statistical parsing

```
A01:0010b - AT     The       the      [O[S[Nns:s.
A01:0010c - NP1s   Fulton    Fulton   [Nns.
A01:0010d - NNL1cb County    county   .Nns]
...
```

(SUSANNE — Sampson)

```
( (S
    (NP-SBJ-23
      (NP (DT No) (NN price) )
      (PP (IN for)
        (NP (DT the) (JJ new) (NNS shares) )))
    (VP (VBZ has)
      (VP (VBN been)
        (VP (VBN set)
          (NP (-NONE- *-23) ))))
    (. .) ))
```

(Penn Treebank — Marcus et al.)

Augmenting Grammars with Probabilities

Two major problems with phrase-structure grammars parsing real-world text: under-generation and ambiguity resolution

- *Time flies like an arrow*, with *time* V, N; *flies* V, N; *like* V, P: 4 parses
- compound nouns, e.g. *airport long term car park courtesy vehicle pick-up point*: N → N N = Catalan series (1430 analyses)
- multiple prepositional phrases, complements vs. adjuncts, etc.

Deliberately have a 'loose' grammar — to deal to some extent with under-generation — and to deal with looseness and ambiguity train a probabilistic model on treebank analyses compatible with grammar

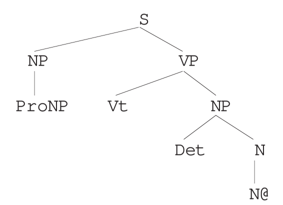
Probabilistic context-free grammar (PCFG)

Associate a probability with each context-free grammar rule $P(X \rightarrow Y...)$, s.t. all $P(X \rightarrow ...)$ sum to 1 for each X

Each derivation of a sentence is assigned a probability = product of probabilities of rules utilised (assume independence). Probability of a sentence = sum of derivation probabilities. Sum of probabilities of all possible sentences is 1 (for any 'reasonable grammar')

Use probabilities to disambiguate / rank parses — able to prefer more common constructions over less common ones

```
1) S' → S (1.0)       6) NP → Det N (0.3)
2) S → NP VP (1.0)    7) NP → NP PP (0.3)
3) VP → Vt NP (0.4)   8) N → N N (0.3)
4) VP → Vi (0.6)      9) PP → P NP (1.0)
5) NP → ProNP (0.4)  10) N → N@ (0.7)
```




```
0.4 x 1.0 x 0.4 x 0.3 x 0.7 = 0.0336
```

To train, extract rules from treebank and estimate their probabilities from frequencies observed in treebank. For Penn Treebank, results in decent accuracy

Grammar induction from unannotated corpora is possible is in principle with PCFG (using variation of HMM training). But several problems:

- very slow
- local maxima
- in practice needs more non-terminals than theoretically needed
- increases probability of training corpus, but no guarantee that structures learned will have any correspondance to linguistic theory; even if initialised with a grammar, training might change meaning of non-terminal categories

Parsing using Contextual Probabilities

Some problems with PCFG:

- PCFG cannot distinguish probabilistically between different derivations which use the same rules
- probabilities of sub-trees cannot be dependent on context (a given rule may be more likely to expand a given type of constituent in one location than another)
- the probability of a smaller tree normally greater than probability of a larger tree

‘Extended-context’ PCFG

Can associate different probabilities with a production depending on the particular nonterminal occurrence (on the RHS of a rule) being rewritten

- probabilities depend on the context within a production
- 23% decrease in incorrect analyses reported over a standard PCFG

Can compute ‘weight’ of of a derivation as mean of the product of the conditional probabilities of each CFG rule appearing in the parse tree given the

- non-terminal trigram context of its dominating rule, and
- the part-of-speech trigram at that point

Aproaches able to distinguish (to a limited extent) probabilities of different trees in which the same rules are applied but in different configurations

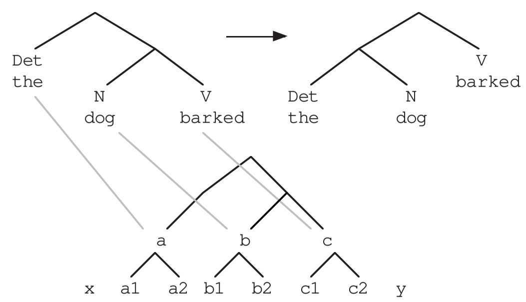
History-based grammar

CFG + conditional probability of rule application:

- P(rule application) depends on word / tag / label at this / parent / sibling / child node
- decision tree ‘learning’ $\rightarrow$ equivalence classes over contexts
- select 1 candidate parse, then refine it
- improvement in most probable parse vs. correct parse from 60% (PCFG) to 75% (paper by Magerman describing system at *ACL-95* conference)

Transformation-based learning of phrase structure

First tag words in sentence and make initial guess of completely right-branching parse tree. Then apply rules which ‘flip’ branches in particular local contexts, e.g.



‘IF a pre-terminal matches the *Det* tag in the *a2* (/*a*) position of a right-branching tree configuration THEN perform a right-to-left flipping transformation’

Probabilistic left corner / LR parsing

Consider probabilities on parser actions when constructing a *derivation*

In *predictive left-corner parsing*, natural to condition parsing actions on left corner and top goal category of parser stack

For shift-reduce parsing controlled by an *LR parse table*, different actions in table correspond to different parse contexts — associate probability with each action by training on a treebank (described in article by Briscoe & Carroll, *Computational Linguistics*, 19:1, 1993)

Parsing using Lexical Probabilities

Stochastic Lexicalised Tree-Adjoining Grammar (SLTAG)

An LTAG consists of multiple-level parse tree fragments (each *anchored* by a lexical item)

Two types of approach

- supertagging model: probabilities on selecting a given tree (= *supertag*) for a particular lexical item, and on tree n-grams; tag text with supertags then glue trees together to form analysis (c.f. part-of-speech tagging)
- stochastic derivation model: each lexicalised tree has a probability of application / a probability is associated with a particular set of *substitutions* into a tree (c.f. PCFG and extensions)

Data-Oriented Parsing (DOP)

Parses directly from treebank trees

Uses subtrees of arbitrary depth occurring in the corpus as sub-analyses, and, given a string of words to parse, 'glues' the sub-analyses together to make full parse trees

- identical analyses derived in many different ways
- $P(derivation)$ = product of sub-analysis probabilities; $P(tree) = \Sigma P(derivation_i)$
- can't use Viterbi: use an iterative Monte Carlo search — estimates most probable parse with an error than can be made arbitrarily small
- good accuracy in limited domains
- very slow

Parsing with bi-lexical dependencies

Good results using a 'generative' language model: start with a parent node and a head lexical item and then model successive generation of dependents to left and to right

Probability of each dependent assumed to be independent; conditional on

- parent grammatical category,
- head node's category and actual lexical item, and
- a function encoding distance, intervening words and punctuation

See *ACL-97* paper by Collins

Probabilistic predicate-argument structure

More realistic model since abstracts away from syntax, but more parameters. Also, training material much harder to construct/obtain

Evaluating Parsers

Evaluating with respect to what? — treebank trees not used in training

Exact match of a tree to a test corpus tree too strict

'PARSEVAL' scheme is a relaxation of exact match to measuring *similarity* of a tree to a test corpus tree. Computes:

- number of bracketing matches M with respect to the number of bracketings P returned by the parser (expressed as precision M/P), and
- with respect to the number C in the corpus (expressed as recall M/C); also
- number of 'crossing' brackets per sentence (where a bracketed sequence from parser overlaps with one from treebank and neither is properly contained in other)

Evaluation on accuracy of identifying dependency relations more reliable?