Language and Computation

week 9, Thursday, March 27

Tamás Biró
Yale University
`tamas.biro@yale.edu`
`http://www.birot.hu/courses/2014-LC/`

Practical matters

- **Post-reading:** JM 12, JM 13, 14.1-4.
- **Pre-reading:** JM 11.1 and intro of 11.3.
- `http://birot.hu/courses/2014-LC/readings.txt`
- Assignment 3 returned. Midterm due now.
- Assignment 4 posted during the weekend. Due: 04/08.
- (To come: Viterbi and Forward-Backward – an example)

Today

- Remarks on Assignment 3 (also as an intro to PCFGs)
- Probabilistic Context Free Grammars
- Parsing PCFGs
- Learning PCFGs
- (Further parsing techniques)

Next time: Computational Phonology

Minimal Cost Paths in a Weighted FSA

Minimal Cost Paths in a Weighted FSA

Weighted Finite State Automata (WFSAs)
vs. Hidden Markov Models (HMMs):

- Σ is finite in WFSA, not necessarily in HMM.
- HMM is multiplicative, WFSA is additive.
- HMM maximizes probabilities, WFSA minimizes costs.
- $-\log$ of HMM probabilities $\rightarrow$ WFSA costs.

Minimal Cost Paths in a Weighted FSA

Weighted Finite State Automata (WFSAs)
vs. Hidden Markov Models (HMMs):

- HMM: transition from any q to any q' possible,
- . . . but maybe $a_{q,q'} = 0$, and so probability of path $= 0$.
- WFSA: transition possible only if $q' \in \delta(q, i)$.
- If goal is minimal cost, then suppose $+\infty$ cost.

Minimal Cost Paths in a Weighted FSA

Weighted Finite State Automata (WFSAs)
vs. Hidden Markov Models (HMMs): **different picture**

- HMM: emission in states.
- WFSA: reading during transitions.
- HMM: stochastic/probabilistic process.
- WFSA: non-deterministic, but not stochastic/probabilistic: transition is either possible, or not.

Probabilistic Context Free Grammars

Probabilistic Context Free Grammars

N a set of **non-terminal symbols** (or **variables**)

Σ a set of **terminal symbols** (disjoint from N)

R a set of **rules** or productions, each of the form $A \rightarrow \beta\ [p]$,
where A is a non-terminal,
β is a string of symbols from the infinite set of strings $(\Sigma \cup N)*$,
and p is a number between 0 and 1 expressing $P(\beta|A)$

S a designated **start symbol**

$$\sum_{\beta \in (N \cup \Sigma)^*} P(A \rightarrow \beta) = 1$$

Probabilistic Context Free Grammars

Independence assumption: rewrite rules are applied independently from each other.

Probability of tree T (which yields sentence S):

$$P(T,S) = \prod_{i=1}^{n} P(RHS_i|LHS_i)$$

the product of the probabilities of the n rules used to expand each of the n non-terminal nodes in parse tree T
(J&M 14.1.1).

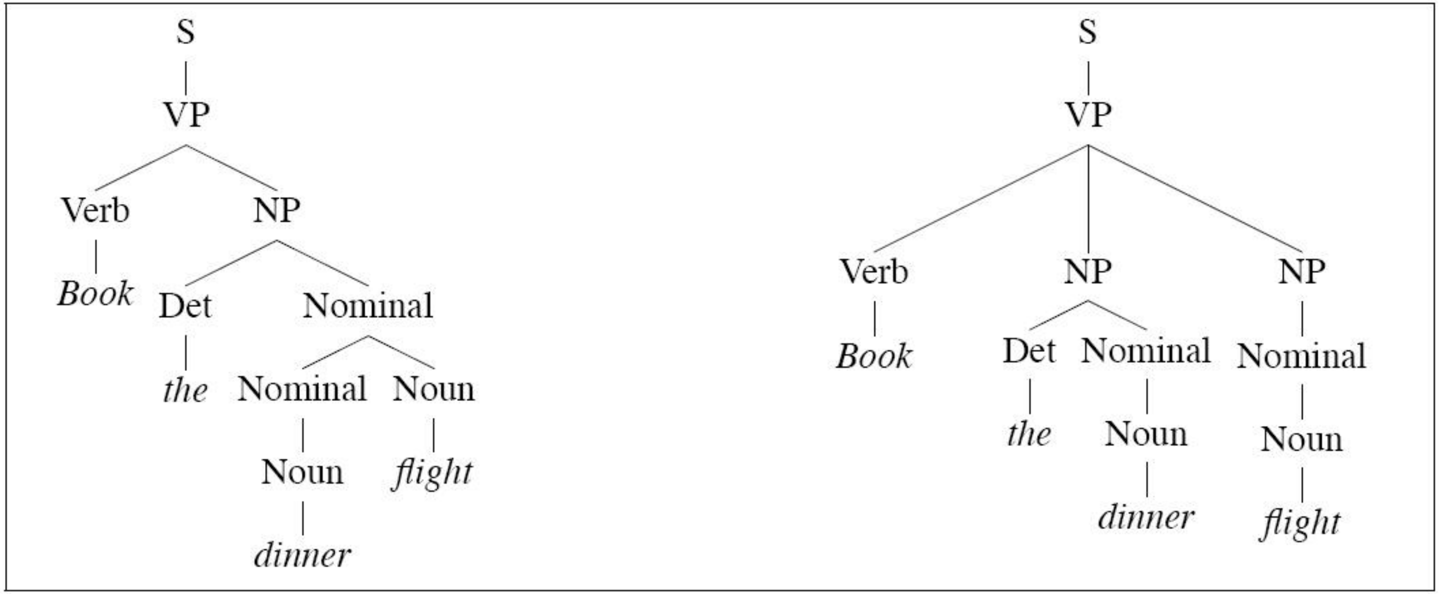
Consistency: if $\sum_{T,S} P(T,S) = 1$. Not always the case!

Probabilistic CFG: an example

Grammar		**Lexicon**
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.15] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flights$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$; [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| NWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

Probabilistic Context Free Grammars

An example:



(booking a flight serving dinner vs. booking a flight on behalf of 'dinner'.)

$P(T_l, S) = 2.2 \times 10^{-6}$ $\qquad$ $P(T_r, S) = 6.1 \times 10^{-7}$

Probabilistic Context Free Grammars

Parse tree T *over* sentence S, a.k.a. S is the *yield* of tree T.

Parse selection: picking the most probable parse:

$$\hat{T}(S) = \operatorname*{argmax}_{T \text{such that } S = \text{yield}(T)} P(T|S)$$

$$\hat{T}(S) = \operatorname*{argmax}_{T \text{s.t. } S = \text{yield}(T)} \frac{P(T, S)}{P(S)}$$

$$\hat{T}(S) = \operatorname*{argmax}_{T \text{s.t. } S = \text{yield}(T)} P(T, S)$$

Parsing and learning a PCFG

Parsing and Learning a PCFG

- **Parsing:** Given (probabilistic) CFG G, given sentence s, find (possible/most probable) parse(s) tree for s in G.

- **Learning:** Given set of (parsed/unparsed) sentences, build a (probabilistic) context free grammar.

Learning a PCFG

- **Learning:** Given set of sentences, build a PCFG.
- **Tree bank:** a set of parsed sentences.
- Maximum likelihood estimate:

$$P(\alpha \to \beta | \alpha) = \frac{\#\alpha \to \beta}{\sum_\gamma \#\alpha \to \gamma} = \frac{\#\alpha \to \beta}{\#\alpha}$$

- Without a tree-bank: **inside-outside algorithm**
  a version of EM, similar to forward-backward for HMM.

Parsing a PCFG

- **Parsing:** Given PCFG G, given sentence s, find (possible/most probable) parse(s) tree for s in G.
- Probabilistic CKY (bottom-up, left-to-right; requires CNF)

```
function PROBABILISTIC-CKY(words,grammar) returns most probable parse
                                                   and its probability
  for j ← from 1 to LENGTH(words) do
     for all { A | A → words[j] ∈ grammar}
        table[j−1, j,A] ← P(A → words[j])
     for i ← from j−2 downto 0 do
        for k ← i+1 to j−1 do
              for all { A | A → BC ∈ grammar,
                          and table[i,k,B] > 0 and table[k, j,C] > 0 }
                 if (table[i,j,A] < P(A → BC) × table[i,k,B] × table[k,j,C]) then
                    table[i,j,A] ← P(A → BC) × table[i,k,B] × table[k,j,C]
                    back[i,j,A] ← {k,B,C}
     return BUILD_TREE(back[1, LENGTH(words), S]), table[1, LENGTH(words), S]
```

See you next week!