November 4, 2003

Ch 12

Probabilistic Parsing

Why probabilistic parsing?

- Ambiguity resolution
- Best-first search
- Modeling human processing (computational psycholinguistics)
- Robustness
- Ambituigy resolution with robust grammars

PCFGs

- $G = (N, \Sigma, P, S, D)$
- N: A set of non-terminal symbols
- Σ: A set of terminal symbols (disjoint from N)
- P: A set of productions (or phrase structure rules)
  $A \rightarrow \ \beta$ where $A \in N$ and $\beta \in (\Sigma \cup N)*$
- S: A desginated start symbol, selected from N.
- D: a function assigning probabilities to each rule in P.

A closer look at D

- Domain: rules of the grammar (P)
- Range: probabilities p (values between 0 and 1)
- For each non-terminal in N, the probabilities of all the rules rewriting N must sum to 1.
- Formallym each p is a conditional probability:

  $$P(A \rightarrow \ \beta \mid A)$$

Sample grammar

S $\rightarrow$ NP VP	[.80]	Det $\rightarrow$ that [.05] | the [.80] | a [.15]	
S $\rightarrow$ Aux NP VP	[.15]	Noun $\rightarrow$ book	[.10]
S $\rightarrow$ VP	[.05]	Noun $\rightarrow$ flights	[.50]
NP $\rightarrow$ Det Nom	[.20]	Noun $\rightarrow$ meal	[.40]
NP $\rightarrow$ Proper-Noun	[.35]	Verb $\rightarrow$ book	[.30]
NP $\rightarrow$ Nom	[.05]	Verb $\rightarrow$ include	[.30]
NP $\rightarrow$ Pronoun	[.40]	Verb $\rightarrow$ want	[.40]
Nom $\rightarrow$ Noun	[.75]	Aux $\rightarrow$ can	[.40]
Nom $\rightarrow$ Noun Nom	[.20]	Aux $\rightarrow$ does	[.30]
Nom $\rightarrow$ Proper-Noun Nom	[.05]	Aux $\rightarrow$ do	[.30]
VP $\rightarrow$ Verb	[.55]	Proper-Noun $\rightarrow$ TWA	[.40]
VP $\rightarrow$ Verb NP	[.40]	Proper-Noun $\rightarrow$ Denver	[.60]
VP $\rightarrow$ Verb NP NP	[.05]	Pronoun $\rightarrow$ you [.40] | I [.60]	

Using the probabilities

- Estimate the joint probability of a parse tree and a sentence:

$$P(T,S) = \prod_{n \in T} p(r(n))$$

- Joint probability = the probability of the parse:

$$P(T,S) = P(T)P(S \mid T) \quad \text{def of joint probability}$$

$$P(S \mid T) = 1 \quad \text{the parse tree includes}$$

$$P(T,S) = P(T) \quad \text{the sentence}$$

- $\rightarrow$ parse selection: $\hat{T}(S) = \underset{T \in \tau(S)}{\text{argmax}}\, P(T \mid S)$

Using the probabilities

- $\hat{T}(S) = \underset{T \in \tau(S)}{\operatorname{argmax}} P(T \mid S)$
- $P(T \mid S) = \frac{P(T,S)}{P(S)}$
- $\hat{T}(S) = \underset{T \in \tau(S)}{\operatorname{argmax}} \frac{P(T,S)}{P(S)}$
- $P(S)$ will be constant, if we're considering the parses of one sentence.
- $\hat{T}(S) = \underset{T \in \tau(S)}{\operatorname{argmax}} P(T)$

Using the probabilities II

- Estimate the probability of a string of words constituting a sentence:
  - Unambiguous strings: $P(T)$
  - Ambiguous strings: $\sum_{T \in \tau(S)} P(T)$
- $\rightarrow$ language modeling in speech recognition
- Probability that a string is a *prefix* of a sentence generated by the grammar (Stolcke 1995), also useful in speech recognition.

Where do the probabilities come from?

- From a treebank, whose trees (can be made to) correspond to the grammar.

$$P(\alpha \rightarrow \ \beta \mid \alpha) = \frac{Count(\alpha \rightarrow \ \beta)}{\Sigma_{\gamma} Count(\alpha \rightarrow \gamma)} = \frac{Count(\alpha \rightarrow \beta)}{Count(\alpha)}$$

- By parsing a corpus, and counting rule occurrences as weighted by the probability of each parse – do this iteratively with the **Inside-Outside** algorithm.

A Bottom-Up Chart Parser (CKY)

for $(0 \leq i \leq length(input))$ do

$chart_{[i,i+1]} \leftarrow \{\alpha \mid \alpha \rightarrow input_i\};$

for $(0 \leq l < input)$ do

for $(0 \leq i < length(input) - 1)$ do

for $(1 \leq j \leq l)$ do

if $(\alpha \rightarrow \beta_1\beta_2 \in P \wedge$

$\beta_1 \in chart_{[i,i+j]} \wedge \beta_2 \in chart_{[i+j,i+l+1]})$ then

$chart_{[i,i+l+1]} \leftarrow chart_{[i,i+l+1]} \cup \{\alpha\};$

Probabilistic CKY

function CKY(*words*, *grammar*) **returns** most probable parse w/probability

Create and clear π[*#words*,*#words*,*#non-terms*]
for $i \leftarrow$ 1 to *#words*
for $A \leftarrow$ 1 to *#non-terms*
if ($A \rightarrow w_i$ is in *grammar*) then
$\pi[i, i, A] \leftarrow P(A \rightarrow w_i)$
for *span* $\leftarrow$ 2 to *#words*
for *begin* $\leftarrow$ 1 to *#words* $-$ *span* +1
end $\leftarrow$ *begin* + *span* $-$ 1
for $m \leftarrow$ *begin* to *end* $-$ 1
for $A, B, C \leftarrow$ 1 to *#non-terms*
prob = π[*begin*,*m*,*B*] × π[$m+1$,*end*,C] × $P(A \rightarrow BC)$
if (*prob* $>$ π[*begin*,*end*,*A*]) then
π[*begin*,*end*,*A*] = *prob*
back[*begin*,*end*,*A*] = {*m*,*B*,*C*}
return BUILD_TREE(*back*[1,*#words*,1]), π[1,*#words*,1]

Summary

- Probabilistic CFGs
- Uses of probabilities
- Learning probabilities
- Probabilistic chart parsing