November 4, 2004

Ch 12

Probabilistic Parsing

Outline

- Why probabilistic parsing?
- Probabilistic CFGs
- Uses of probabilities
- Learning probabilities
- Probabilistic chart parsing
- Midterm

Why parsing?

- Linguistic research
- Natural language understanding systems
- Language modeling for speech recognition (possibly)
- Machine translation (possibly)
- Because it’s there...

Statistical methods in NLP/speech

- Apply machine learning techniques to linguistic problems
- Work from large data set (corpora, treebanks, . . .)
- Supervised or unsupervised
- Tend to be “robust”: come up with an answer for everything (or multiple ranked answers).
- Strive for portability across languages and domains
- Sort of like origami...
- We’ll see just one example in this course: PCFGs
- Modern successful systems merge stochastic and symbolic techniques.

Why probabilistic parsing?

- Ambiguity resolution
- Best-first search
- Modeling human processing (computational psycholinguistics)
- Robustness
- Ambiguity 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)
  $A \longrightarrow \ \beta$ where $A \in N$ and $\beta \in (\Sigma \cup N)*$
- S: A designated 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.
- Formally each p is a conditional probability:

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

Sample grammar

S $\longrightarrow$ NP VP	[.80]
S $\longrightarrow$ Aux NP VP	[.15]
S $\longrightarrow$ VP	[.05]
NP $\longrightarrow$ Det Nom	[.20]
NP $\longrightarrow$ Proper-Noun	[.35]
NP $\longrightarrow$ Nom	[.05]
NP $\longrightarrow$ Pronoun	[.40]
Nom $\longrightarrow$ Noun	[.75]
Nom $\longrightarrow$ Noun Nom	[.20]
Nom $\longrightarrow$ Proper-Noun Nom	[.05]
VP $\longrightarrow$ Verb	[.55]
VP $\longrightarrow$ Verb NP	[.40]
VP $\longrightarrow$ Verb NP NP	[.05]

Det $\longrightarrow$ that [.05] | the [.80] | a [.15]	
Noun $\longrightarrow$ book	[.10]
Noun $\longrightarrow$ flights	[.50]
Noun $\longrightarrow$ meal	[.40]
Verb $\longrightarrow$ book	[.30]
Verb $\longrightarrow$ include	[.30]
Verb $\longrightarrow$ want	[.40]
Aux $\longrightarrow$ can	[.40]
Aux $\longrightarrow$ does	[.30]
Aux $\longrightarrow$ do	[.30]
Proper-Noun $\longrightarrow$ TWA	[.40]
Proper-Noun $\longrightarrow$ Denver	[.60]
Pronoun $\longrightarrow$ 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:

$P(T,S) = P(T)P(S \mid T)$	def of joint probability
$P(S \mid T) = 1$	the parse tree includes
$P(T,S) = P(T)$	the sentence

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

Using the probabilities

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

Using the probabilities II

- Estimate the probability of a string of words constituting a sentence:
  - Unambiguous strings: $P(T)$
  - Ambiguous strings: $\sum_{T \in \tau(S)} P(T)$
- $\longrightarrow$ 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.

Another Chart Parser (CKY)

Create and clear *chart*[#*words*,#*words*]

for $i \leftarrow$ 1 to #*words*

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

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

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

$\beta_1 \in chart_{[begin,m]} \wedge \beta_2 \in chart_{[m+1,end]})$ then

$chart_{[begin,end]} \leftarrow chart_{[begin,end]} \cup \{\alpha\};$

Probabilistic CKY

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

    Create, clear π[*#words*,*#words*,*#non-terms*], *back*[*#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*] $\times$ π[$m+1$,*end*,C] $\times$ $P(A \rightarrow BC)$
                    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
- Next time: inside-outside, problems with PCFGs, probabilistic lexicalized CFGs, evaluating parsers
- Now: on to the midterm