

CSE842: Natural Language Processing

Lecture 11: Probabilistic Parsing (II)

2/18/2009

CSE842, Spring 2009, MSU

1

Probabilistic Context Free Grammars

- The simplest augmentation of the Context Free Grammar (CFG) is the Probabilistic Context Free Grammar (PCFG)
- CFG is defined by (N, Σ, P, S)
 - N : a set of non-terminals
 - Σ : set of terminals
 - P : set of productions of the form: $A \rightarrow \beta$, $A \in N$, β a string of symbols $s \in (\Sigma \cup N)$
 - S : a designated start symbol

2/18/2009

CSE842, Spring 2009, MSU

2

PCFG Augmentation

- Each rule in P is assigned a conditional probability $A \rightarrow \beta | p$
- A PCFG is a 5 tuple $G=(N, \Sigma, P, S, D)$ where D is a function assigning probabilities to each rule in P $P(A \rightarrow \beta)$ or $P(A \rightarrow \beta | A)$
- Given a non-terminal A , and r_1, r_2, \dots, r_i , all expanding A

$$\sum_{j=1}^i P(r_j) = 1$$

2/18/2009

CSE842, Spring 2009, MSU

3

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]
$NP \rightarrow Proper-Noun$	[.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]	$Proper-Noun \rightarrow Houston$	[.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]		

2/18/2009

CSE842, Spring 2009, MSU

4

Using Probabilities

- To estimate probabilities concerning a sentence and its parse trees
- Useful in disambiguation
- How can we define the probability of a tree?

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

$$P(T, S) = P(T) P(S | T) = P(T)$$

2/18/2009

CSE842, Spring 2009, MSU

5

Use in Disambiguation

- The most likely tree for a sentence S

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T | S)$$

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} \frac{P(T, S)}{P(S)}$$

$$\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T, S)$$

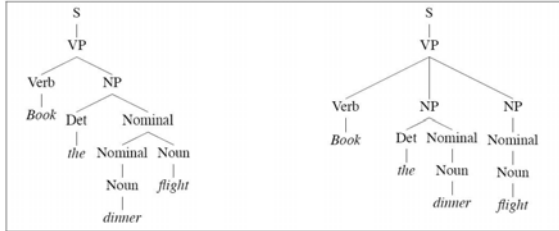
But $P(T, S) = P(T)$ \rightarrow $\hat{T}(S) = \arg \max_{T \in \tau(S)} P(T)$

2/18/2009

CSE842, Spring 2009, MSU

6

Disambiguation Example



2/18/2009

CSE842, Spring 2009, MSU

7

Disambiguation Example

Rules	P	Rules	P
S → VP	.05	S → VP	.05
VP → Verb NP	.20	VP → Verb NP NP	.10
NP → Det Nominal	.20	NP → Det Nominal	.20
Nominal → Nominal Noun	.20	NP → Nominal	.15
Nominal → Noun	.75	Nominal → Noun	.75
		Nominal → Noun	.75
Verb → book	.30	Verb → book	.30
Det → the	.60	Det → the	.60
Noun → dinner	.10	Noun → dinner	.10
Noun → flights	.40	Noun → flights	.40

$$P(T_{\text{left}}) = 0.05 * 0.2 * 0.2 * 0.2 * 0.75 * 0.3 * 0.6 * 0.1 * 0.4 = 2.2 \times 10^{-6}$$

$$P(T_{\text{right}}) = 0.05 * 0.1 * 0.2 * 0.15 * 0.75 * 0.75 * 0.3 * 0.6 * 0.1 * 0.4 = 6.1 \times 10^{-7}$$

2/18/2009

CSE842, Spring 2009, MSU

8

Learning PCFG Probabilities

Given a set of examples trees, use MLE

$$P(\alpha \rightarrow \beta | \alpha) = \frac{\text{Count}(\alpha \rightarrow \beta)}{\sum_{\gamma} \text{Count}(\alpha \rightarrow \gamma)} = \frac{\text{Count}(\alpha \rightarrow \beta)}{\text{Count}(\alpha)}$$

2/18/2009

CSE842, Spring 2009, MSU

9

Algorithms for PCFG

Given a PCFG and a sentence S , define $\tau(S)$ to be the set of trees with S as the yield.

- How do we find the best parse tree for the sentence S ?

$$\arg \max_{T \in \tau(S)} P(T, S)$$

- How do we find the probability of S ?

$$P(S) = \sum_{T \in \tau(S)} P(T, S)$$

2/18/2009

CSE842, Spring 2009, MSU

10

Review: Chomsky Normal Form

A context free grammar $G = (N, \Sigma, P, S)$ in Chomsky Normal Form is as follows

- N is a set of non terminal symbols
- Σ is a set of terminal symbols
- P is a set of rules which take one of two forms
 - $X \rightarrow Y_1 Y_2$ for $X \in N$, and $Y_1, Y_2 \in N$
 - $X \rightarrow Y$ for $X \in N$, and $Y \in \Sigma$
- $S \in N$ is a distinguished start symbol

2/18/2009

CSE842, Spring 2009, MSU

11

Probabilistic Parsing

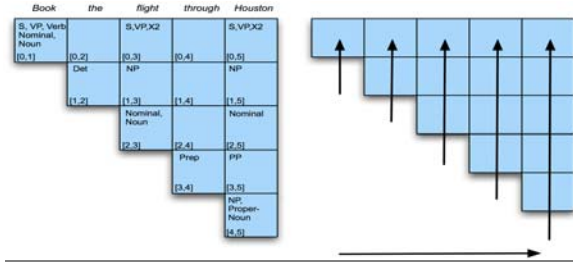
- CYK Algorithm: bottom up parser
- Input:
 - A Chomsky normal form PCFG, $G = (N, \Sigma, P, S, D)$ Assume that the K non-terminals have indices $1, 2, \dots, K$, and the start symbol S has index 1
 - n words w_1, \dots, w_n
- Data Structure:
 - A dynamic programming array $\pi[i, j, k]$ holds the **maximum probability** for a constituent with non-terminal index N_k spanning words $i..j$.
- Output: The maximum probability parse $\pi[0, n, 1]$

2/18/2009

CSE842, Spring 2009, MSU

12

CYK Review



2/18/2009

CSE842, Spring 2009, MSU

13

Base Case

- CYK fills out $\pi[i, j, k]$ by induction
- Base case
 - Input strings with length = 1 (individual words w_i)
 - In CNF, the probability of a given non-terminal N_k expanding to a single word w_i must come only from the rule $N_k \rightarrow w_i$

For all $j = 1..n$, for $k = 1..K$

$$\pi[j-1, j, k] = P(N_k \rightarrow w_j | N_k)$$

2/18/2009

CSE842, Spring 2009, MSU

14

Recursive Case

- For strings of words $> 1, N_k \Rightarrow^* w_{ij}$ if and only if there is one rule $N_k \rightarrow N_l N_m$ and $s, i \leq s \leq j$ such that N_l derives the first $s-i$ symbols and N_m derives the last $j-s$ symbols.
- Since w_{is} and w_{sj} are shorter than w_{ij} , $\pi[i, s, l]$ and $\pi[s+1, j, m]$ are already stored.

$$\pi[i, j, k] = \pi[i, s, l] * \pi[s+1, j, m]$$

for all $j = 1..n, i = (j-2)$ downto 0, $k = 1..K,$

$$\pi[i, j, k] = \max_{\substack{l \leq s < j \\ l \leq l \leq K \\ l \leq m \leq K}} \{ P(N_k \rightarrow N_l N_m | N_k) \times \pi[i, s, l] \times \pi[s+1, j, m] \}$$

2/18/2009

CSE842, Spring 2009, MSU

15

CYK Algorithm for Probabilistic Parsing

Initialization

for $j = 1..n, k = 1..K$

$$\pi[j-1, j, k] = P(N_k \rightarrow w_j | N_k)$$

Main Loop:

for $j = 1..length$

for $i = j-2$ downto 0

for $k = 1..K$

max = 0;

for $s = (i+1)..(j-1), l = 1..K, m = 1..K,$

prob = $P(N_k \rightarrow N_l N_m | N_k) \times \pi[i, s, l] \times \pi[s+1, j, m]$

if prob > max

max = prob;

split $(i, j, k) = (s, l, m);$ //store backpointers to indicate the best split

$\pi[i, j, k] = max$

2/18/2009

CSE842, Spring 2009, MSU

16

Problem with PCFG

- Lack of sensitivity to structural dependency
- Lack of sensitivity to lexical dependency

2/18/2009

CSE842, Spring 2009, MSU

17

Structure Dependency

- Each PCFG rule is assumed to be independent of each other rule, and thus the rule probabilities are multiplied together.
- Observation: sometimes the choice of how a node expands is dependent on the location of the node in the parse tree
 - NP \rightarrow Pronoun, NP \rightarrow Det Noun depends on whether the NP was a subject or an object

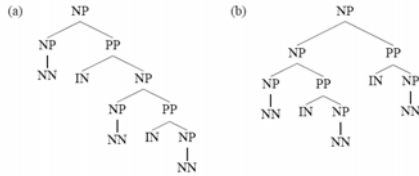
2/18/2009

CSE842, Spring 2009, MSU

18

Structure Dependency

- President of MSU in America



- Both structures have the same rules and therefore receive an equal probability under PCFG.
- However, "close attachment" (structure a) are twice as likely in Wall Street Journal.

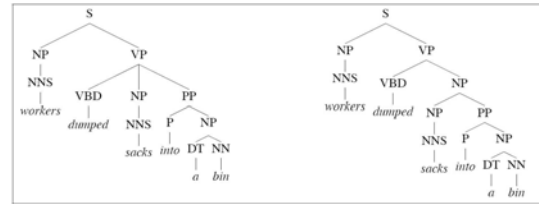
2/18/2009

CSE842, Spring 2009, MSU

19

Lexical Dependency

- Workers dumped sacks into a bin

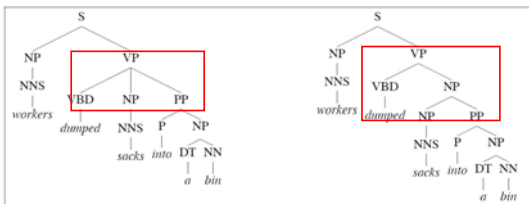


2/18/2009

CSE842, Spring 2009, MSU

20

Lexical Dependency



- (a): $VP \rightarrow VBD\ NP\ PP$
- (b): $VP \rightarrow VBD\ NP; NP \rightarrow NP\ PP$
- Depending on these probabilities, a PCFG will always either prefer NP attachment or VP attachment (e.g., NP attachment is slightly more often)

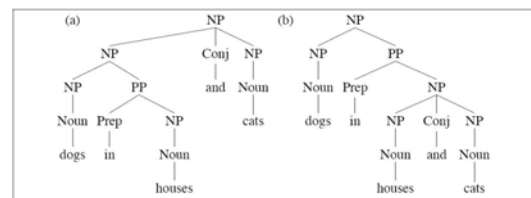
2/18/2009

CSE842, Spring 2009, MSU

21

Lexical Dependency

- dogs in houses and cats



Same probabilities for these two structures

2/18/2009

CSE842, Spring 2009, MSU

22

Dealing with Structure Dependency

- How to deal with structure dependency? For example, modeling the fact that NPs in subject position tend to be pronoun and in object position tend to have full lexical form (e.g., Det Nominal).
- Splitting non-terminals
 - E.g., Split NP into two categories: NP_{subject} and NP_{object}
 - NP_{subject} → Pronoun; NP_{object} → Pronoun
 - NP_{subject} → Det Nominal; NP_{object} → Det Nominal;

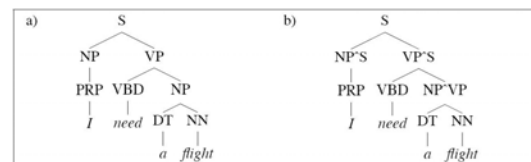
2/18/2009

CSE842, Spring 2009, MSU

23

Parent Annotation

- Annotate each phrasal non-terminals with their parents



Recent work also splitting non-terminal POS nodes (e.g., adverbs)

2/18/2009

CSE842, Spring 2009, MSU

24

Lexicalized PCFG

- Syntactic constituents could be associated with a lexical **head**.
- Lexicalized grammar can be considered as a simple context free grammar with many copies of each rule.
 - One copy corresponds to each possible headword for each constituent.
 - E.g., VP(dumped)->VBD(dumped) NP(sacks) PP(into) $3*10^{10}$

2/18/2009

CSE842, Spring 2009, MSU

25

Heads in CFG

Identify the "head" of each rule

S	⇒	NP	VP
VP	⇒	V _i	
VP	⇒	V _t	NP
VP	⇒	VP	PP
NP	⇒	DT	NN
NP	⇒	NP	PP
PP	⇒	IN	NP

V _i	⇒	sleeps
V _t	⇒	saw
NN	⇒	man
NN	⇒	woman
NN	⇒	telescope
DT	⇒	the
IN	⇒	with
IN	⇒	in

Note: S=sentence, VP=verb phrase, NP=noun phrase, PP=prepositional phrase, DT=determiner, V_i=intransitive verb, V_t=transitive verb, NN=noun, IN=preposition

2/18/2009

CSE842, Spring 2009, MSU

26

Rules to Identify Heads

An example for "NP"

If the rule contains NN, NNS, or NNP:
Choose the rightmost NN, NNS, or NNP

Else If the rule contains an NP: Choose the leftmost NP

Else If the rule contains a JJ: Choose the rightmost JJ

Else If the rule contains a CD: Choose the rightmost CD

Else Choose the rightmost child

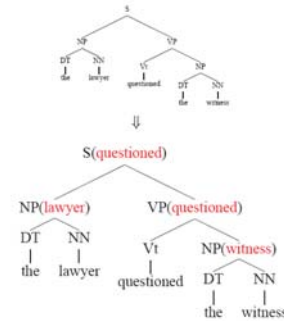
e.g.,
 NP ⇒ DT NNP NN
 NP ⇒ DT NN NNP
 NP ⇒ NP PP
 NP ⇒ DT JJ
 NP ⇒ DT

2/18/2009

CSE842, Spring 2009, MSU

27

Adding Headwords to the Tree

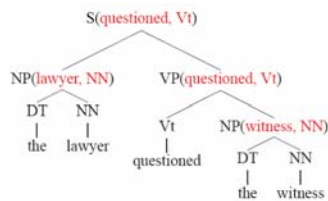


2/18/2009

CSE842, Spring 2009, MSU

28

Adding Headwords to the Tree



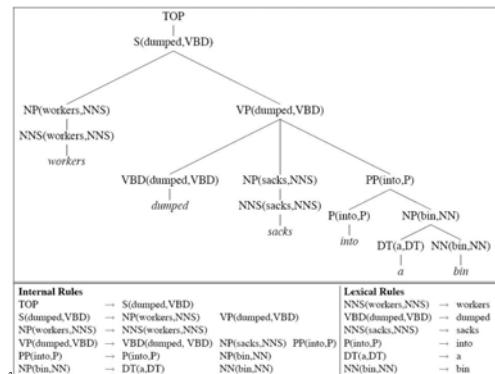
- A constituent receives its head from its head child.
- The POS tags of the headwords are also propagated up the tree.

2/18/2009

CSE842, Spring 2009, MSU

29

An Example



Key Question

- How to estimate the probabilities for internal rules?

$$\frac{\text{Count}(\text{VP}(\text{dumped}, \text{VBD}) \rightarrow \text{VBD}(\text{dumped}, \text{VBD}) \text{NP}(\text{sacks}, \text{NNS}) \text{PP}(\text{into}, \text{P}))}{\text{Count}(\text{VP}(\text{dumped}, \text{VBD}))}$$

- Sparse data problem
- Modern statistical parsers differ in which independence assumptions they make.

2/18/2009

CSE842, Spring 2009, MSU

31

Anatomy of Lexicalized Rules

- An example lexicalized rule:

$\text{VP}(\text{dumped}, \text{VBD}) \rightarrow \text{VBD}(\text{dumped}, \text{VBD}) \text{NP}(\text{sacks}, \text{NNS}) \text{PP}(\text{into}, \text{IN})$

- Each non-terminal is a triple consisting of:

- A label, a word, a tag (i.e., part-of-speech tag of the word)

For example:

for $\text{VP}(\text{dumped}, \text{VBD})$: label = VP, word = dumped, tag = VBD

for $\text{VBD}(\text{dumped}, \text{VBD})$: label = VBD, word = dumped, tag = VBD

2/18/2009

CSE842, Spring 2009, MSU

32

Anatomy of Lexicalized Rules

- An example lexicalized rule:

$\text{VP}(\text{dumped}, \text{VBD}) \rightarrow \text{VBD}(\text{dumped}, \text{VBD}) \text{NP}(\text{sacks}, \text{NNS}) \text{PP}(\text{into}, \text{IN})$

- The **parent** of the rule is the non-terminal on the left-hand-side (LHS) of the rule
 - In this example: $\text{VP}(\text{dumped}, \text{VBD})$
- The **head** of the rule is a single non-terminal on the right-hand-side (RHS) of the rule
 - In this example: $\text{VBD}(\text{dumped}, \text{VBD})$
- The **left-modifiers** of the rule are any non-terminals appearing to the left of the head (can be any number 0 or greater)
 - In this example, no left-modifiers
- The **right-modifiers** of the rule are any non-terminals appearing to the right of the head (can be any number 0 or greater)
 - In this example, two right modifiers: $\text{NP}(\text{sacks}, \text{NNS})$ and $\text{PP}(\text{into}, \text{IN})$

2/18/2009

CSE842, Spring 2009, MSU

33

The General Form of a Lexicalized Rule

$X(h, t) \rightarrow L_n(lw_n, lt_n) \dots L_1(lw_1, lt_1) H(h, t) R_1(rw_1, rt_1) \dots R_m(rw_m, rt_m)$

- $X(h, t)$ is the parent of the rule
- $H(h, t)$ is the head of the rule
- There are n left modifiers, $L_i(lw_i, lt_i)$ for $i = 1..n$
- There are m right modifiers, $R_i(rw_i, rt_i)$ for $i = 1..m$
- There can be zero or more left or right modifiers: $n \geq 0$ and $m \geq 0$

2/18/2009

CSE842, Spring 2009, MSU

34

Simplified Collins Parser

$X(h, t) \rightarrow L_n(lw_n, lt_n) \dots L_1(lw_1, lt_1) H(h, t) R_1(rw_1, rt_1) \dots R_m(rw_m, rt_m)$

$\text{VP}(\text{dumped}, \text{VBD}) \rightarrow \text{VBD}(\text{dumped}, \text{VBD}) \text{NP}(\text{sacks}, \text{NNS}) \text{PP}(\text{into}, \text{IN})$

To estimate the probability of the above rule, apply a generative process

- First, given the parent (i.e., LHS $X(h, t)$), generate the head of the rule
- Then generate the modifiers, one by one, from the inside out.

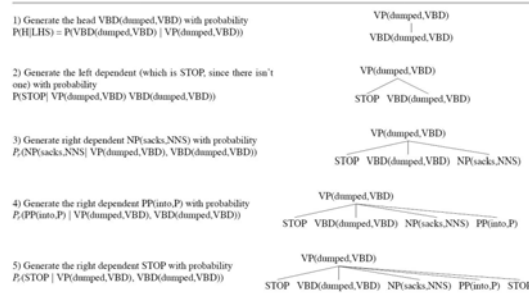
Add STOP at the left and right edges of the rule, which allows the model to know when to stop generating modifiers on a given side.

2/18/2009

CSE842, Spring 2009, MSU

35

An Example



2/18/2009

CSE842, Spring 2009, MSU

36

The Probability of a Rule

$$P(VP(dumped, VBD) \rightarrow VBD(dumped, VBD) NP(sacks, NNS) PP(into, P))$$

Can be estimated as the following using conditional independence assumption:

$$\begin{aligned} & P_H(VBD_{Htable} | VP, dumped, VBD) \\ & \times P_L(STOP | VP, VBD_{Htable}, dumped, VBD) \\ & \times P_R(NP(sacks, NNS) | VP, VBD_{Htable}, dumped, VBD) \\ & \times P_R(PP(into, P) | VP, VBD_{Htable}, dumped, VBD) \\ & \times P_R(STOP | VP, VBD_{Htable}, dumped, VBD) \end{aligned}$$

2/18/2009

CSE842, Spring 2009, MSU

37

Parameter Estimation

Use MLE for parameter estimation, which is less subjective to sparse problem.

$$P_s(NP(sacks, NNS) | VP, VBD, dumped) = \frac{\text{Count}(VP(dumped, VBD) \text{ with } VBD \text{ as the head label and with } NNS(sacks, NNS) \text{ as a child somewhere on the right})}{\text{Count}(VP(dumped, VBD) \text{ with } VBD \text{ as the head label})}$$

Interpolating backed-off models:

Backoff Level	$P_R(R_i(rw_i, rt_i) \dots)$	Example
1	$P_R(R_i(rw_i, rt_i) P, lw, ht)$	$P_R(NP(sacks, NNS) VP, VBD, dumped)$
2	$P_R(R_i(rw_i, rt_i) P, ht)$	$P_R(NP(sacks, NNS) VP, VBD)$
3	$P_R(R_i(rw_i, rt_i) P)$	$P_R(NP(sacks, NNS) VP)$

2/18/2009

CSE842, Spring 2009, MSU

38

Evaluating Parsers

Recall, Precision, and F-measurement

$$\text{Recall} = \frac{\text{\# of correct constituents in hypothesis parse of } s}{\text{\# of correct constituents in reference parse of } s}$$

$$\text{Precision} = \frac{\text{\# of correct constituents in hypothesis parse of } s}{\text{\# of total constituents in hypothesis parse of } s}$$

$$F_\beta = \frac{(\beta^2 + 1)PR}{\beta^2 P + R}$$

Cross-brackets: the number of constituents for which the reference parse has a bracketing such as ((A B) C) but the hypothesis parse has a bracketing such as (A (B C))

2/18/2009

CSE842, Spring 2009, MSU

39