CS371N Lecture 16
Constituency Parsing, CKY

Announcements

- A4 due Tues
- midterm next Thurs

Recap Want to use HMMs to find

$$\underset{\bar{y}}{\operatorname{argmax}}\, P(\bar{y}|\bar{x})$$ (most likely tags for given sentence)

$$= \underset{\bar{y}}{\operatorname{argmax}}\, \frac{P(\bar{y}, \bar{x})}{P(\bar{x})}$$ ← constant w.r.t. $\bar{y}$

Viterbi algorithm

$v_i(t)$ = score of best path ending in t at step i

$$v_1(\tilde{y}) = \log P(x_1|\tilde{y}) + \log P(\tilde{y})$$

$$v_i(\tilde{y}) = \log P(x_i \mid \tilde{y}) + \max_{\tilde{y}_{prev}} \left[\log P(\tilde{y} \mid \tilde{y}_{prev}) + V_{i-1}(\tilde{y}_{prev}) \right]$$

Today

- Constituency syntax: see slides
- PCFGs
- CKY algorithm

Context-free Grammars

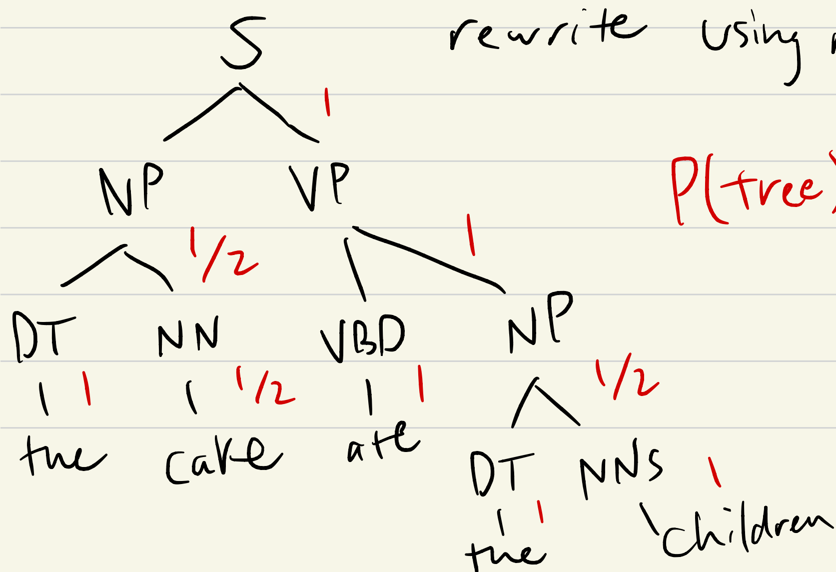
$\{N$	T	S	$R\}$
nonterminals	terminals	start symbol	rules
S, VP, NP, etc.	words	S	

Rules

binary	probs	unary	
S → NP VP	1	DT → the	1
VP → VBD NP	1	NNS → children	1
NP → DT NN	1/2	NN → cake	1/2
NP → DT NNS	1/2	NN → spoon	1/2
		VBD → ate	1

CFGs define a set of trees

rewrite using rules



$P(\text{tree}) = \frac{1}{8}$

Probabilistic context-free grammars (PCFGs)

Each rule has a prob.

Probs. normalize per parent

$P(\text{rule} \mid \text{parent})$

Ex. $P(\text{rule} \mid NP) = \begin{cases} NP \to DT\ NN & 1/2 \\ NP \to DT\ NNS & 1/2 \end{cases}$

$$P(\text{tree}) = \prod_{\text{rules} \in \text{tree}} P(\text{rule} \mid \text{parent})$$

Building a parser

Input: treebank (sents w/ labeled trees)

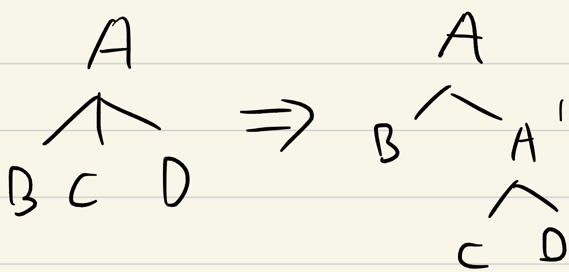
Output: grammar (PCFG)

(also need parsing algo.)

Steps

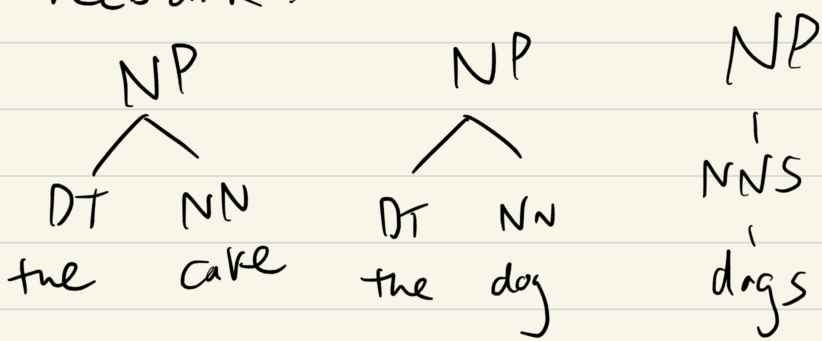
① Grammar preprocessing step

- Make trees have binary + unary rules only



② Read off grammar + compute probs. Count + normalize

Treebank:



$$P(\text{rule} \mid NP) = \begin{cases} NP \to DT\ NN & 2/3 \\ NP \to NNS & 1/3 \end{cases}$$

$$P(\text{word} \mid NN) = \begin{cases} \text{cake} & 1/2 \\ \text{dog} & 1/2 \end{cases}$$

$$P(\text{word} \mid NNS) = \{ \text{dogs} \quad 1$$

similar to HMM emissions

$P(\text{word} \mid \text{tag})$

③ CKY Inputs: PCFG, sentence $\bar{x}$

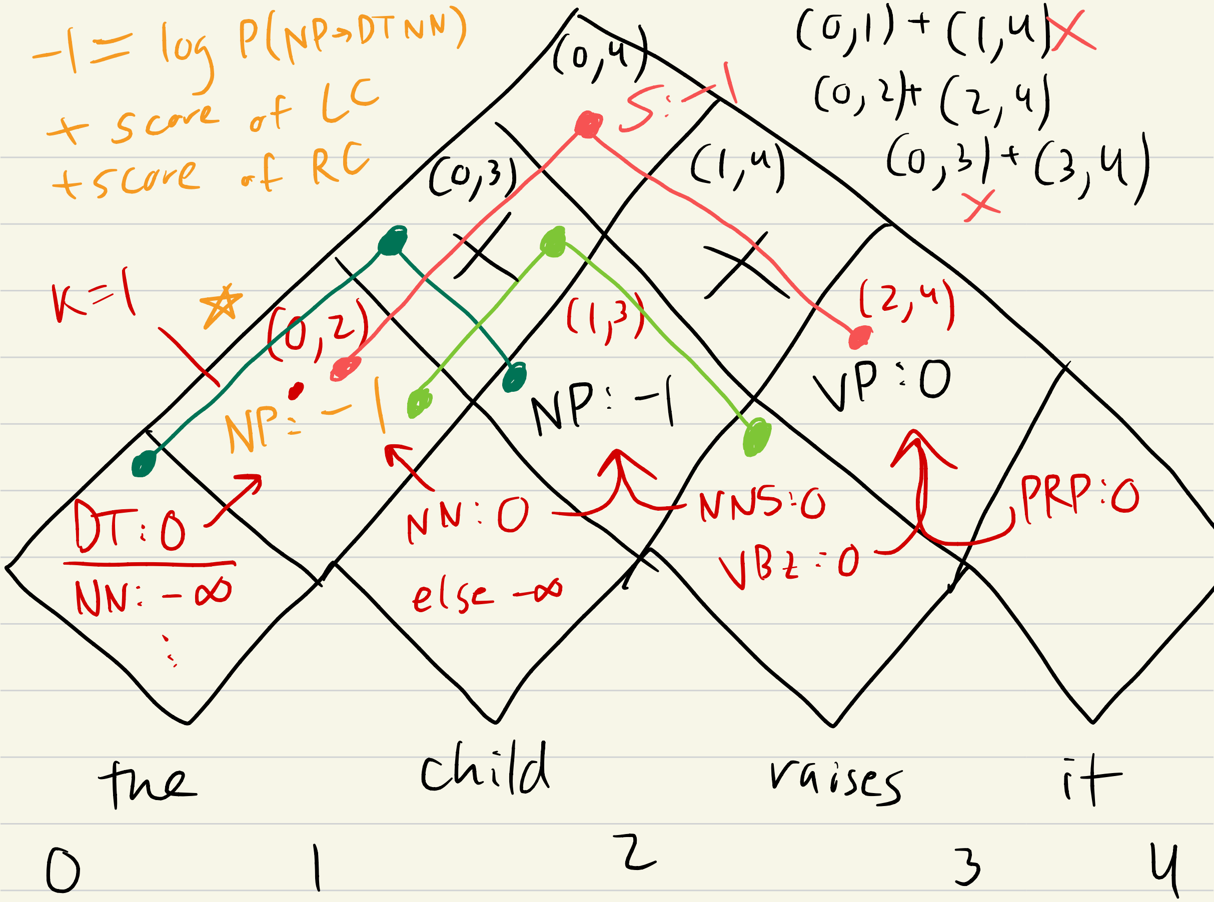
Output: $\underset{T}{\operatorname{argmax}}\ P(T \mid \bar{x})$

most likely tree T for that sentence

Dynamic program: track the best score for building a nonterminal over each span of the sentence $(\approx v_i(\tilde{y}))$

$T(i, j, X)$ = score (log prob) of the best way to build constituent X over span (i, j)
(word i up through word j)

We compute $\underset{T}{\operatorname{argmax}}\ P(T, \bar{x})$



Grammar

DT → the	1	S → NP VP	1
NN → child	1	☆ NP → DT NN	1/2
NNS → raises	1	NP → NN NNS	1/2
VBZ → raises	1	VP → VBZ PRP	1
PRP → it	1		

Assume $\log(1/2) = -1$

runtime: $O(n^3)$
$O(n^3 G)$

CKY:

Base case: $T(i, i+1, X) = \log P(w_i | X)$
(loop over all X)

Recursive case

$T(i, j, X)$ loop over all is + js + Xs
loop over split points k

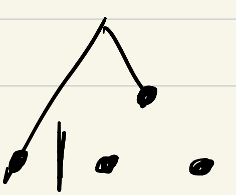
"transition"

$$= \max_{k : i < k < j} \ \max_{X \to X_1 X_2} \left[\log P(X \to X_1 X_2) + T(i, k, X_1) + T(k, j, X_2) \right]$$

Iterate upwards in terms of span length

(0,3)

K=1



K=2