Assignment 3

Manasa Bollavaram

February 23, 2017

1 PCFG Language Models

Comparison between N-gram laguage models and PCFG language models. N-gram language models evaluate the probability of a sentence based on previous n words observed where as PCFG evaluate the probability based on a context free pre assigned grammar rules. One problem with PCFG is the lack of sensitivity to lexical dependency and this is where N-gram model proves to be advantageous because N-gram models take the words into account, especially when the training corpus is large. But, when the training corpus is limited then PCFG can be advantageous because it can be applied to unseen sentences since it operates on grammar rules.

2 Playing with Off-the-shelf Parsers

2.1 Error in PCFG Parsing

- Sentence: *"I saw the boy with the game."*
- Parser used: *http://tomato.banatao.berkeley.edu:8080/parser/parser.html*
- Parse Tree is shown in Fig 1.
- Parsing error: Prepositional Phrase Attachment Ambiguity

The prepositional phrase *"with the game"* should be modifying the noun *"the boy"* instead of the verb *"saw"*.

- Sentence: *"The rich laugh at the poor."*
- Parser used: *http://tomato.banatao.berkeley.edu:8080/parser/parser.html*
- Parse Tree is shown in Fig 2.

Figure 1: Parse Tree: PP attachment ambiguity

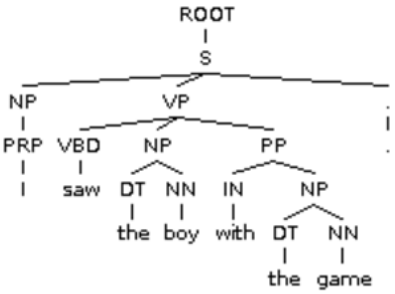
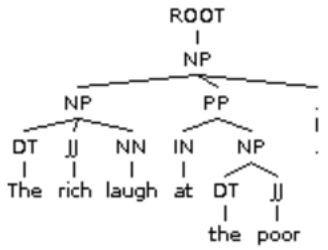


Figure 2: Parse Tree: Lexical Ambiguity



- Parsing error: Lexical Ambiguity

The words *"rich"* and *"poor"* should have been parsed as nouns instead of adjectives.

2.2 Error in Dependency Parsing

- Sentence: *"Shift to your side and roll"*
- Parser used: *http://nlp.stanford.edu:8080/parser/index.jsp*
- Dependency Parse Result:

```
root(ROOT-0, Shift-1)
case(side-4, to-2)
nmod:poss(side-4, your-3)
nmod(Shift-1, side-4)
cc(side-4, and-5)
conj(side-4, roll-6)
```

- Parsing error: *conj(side-4, roll-6) cc(side-4, and-5)*

The parser misidentified the word *"roll"* as noun and therefore the dependencies *"cc"* and *"conj"* established are wrong. The dependency *conj* should be between the verbs *"shift"* and *"roll"*.

- Sentence: *"I convinced her bathroom is messy"*
- Parser used: *http://nlp.stanford.edu:8080/parser/index.jsp*
- Dependency Parse Result:

```
nsubj(convinced-2, I-1)
root(ROOT-0, convinced-2)
nmod:poss(bathroom-4, her-3)
nsubj(messy-6, bathroom-4)
cop(messy-6, is-5)
ccomp(convinced-2, messy-6)
```

- Parsing error: *nmod:poss(bathroom-4, her-3)*

The word *"bathroom"* is considered as object for *"her"* and this is wrongly parsed.

3 Variations to Cocke Younger Kasami (CKY) Algorithm

Pseudo-code developed based on Ashmita Normal Form:

function CKY-ASHMITA-PARSE(words,grammar) **returns** *table*
  for $j \leftarrow$ **from** 1 **to** LENGTH(words) **do**
    $table[j-1, j] \leftarrow \{A | A \rightarrow words[j] \in grammar\}$
    for $i \leftarrow$ **from** $j-2$ **downto** 0 **do**
      for $k \leftarrow i+1$ **to** $j-1$ **do**
        $table[i, j] \leftarrow table[i, j] \cup \{A | A \rightarrow BC \in grammar,$
          $B \in table[i, k], C \in table[k, j]\}$
        for $p \leftarrow k+1$ **to** $j-1$ **do**
          $table[i, j] \leftarrow table[i, j] \cup \{A | A \rightarrow BCD \in grammar,$
            $B \in table[i, k], C \in table[k, p], D \in table[p, j]\}$

4 CFG Grammar Refinement

4.1 a. PCFG derived from the given treebank

S → NP VP [1.0]
NP → John [0.17] | Sally [0.32] | Bill [0.17] | Fred [0.17] | Jeff [0.17]
VP → V1 SBAR [0.33]
VP → VP ADVP [0.33]
VP → snored [0.11] | ran [0.11] | swam [0.11]
V1 → said [0.33] | declared [0.33] | pronounced [0.33]
SBAR → COMP S [1.0]
COMP → that [1.0]
ADVP → loudly [0.33] | quickly [0.33] | elegantly [0.33]

The two rules a) VP→V2 and b) V2→ verb have been modified to be in compliance with Chomsky Normal Form. They have been converted to one rule VP→ verb.

b. CKY Chart and all possible PCFG Parse Trees

CKY table is presented in fig. 3 and the two possible parse trees along with the probability computations are shown in fig. 4 and fig. 5.

Figure 3: CKY Table

Jeff	pronounced	that	Bill	ran	elegantly
NP				S	S
[0,1]	[0,2]	[0,3]	[0,4]	[0,5]	[0,6]
	V1			VP	VP
	[1,2]	[1,3]	[1,4]	[1,5]	[1,6]
		COMP		SBAR	SBAR
		[2,3]	[2,4]	[2,5]	[2,6]
			NP	S	S
			[3,4]	[3,5]	[3,6]
				VP	VP
				[4,5]	[4,6]
					ADVP
					[5,6]

Figure 4: First Parse Tree

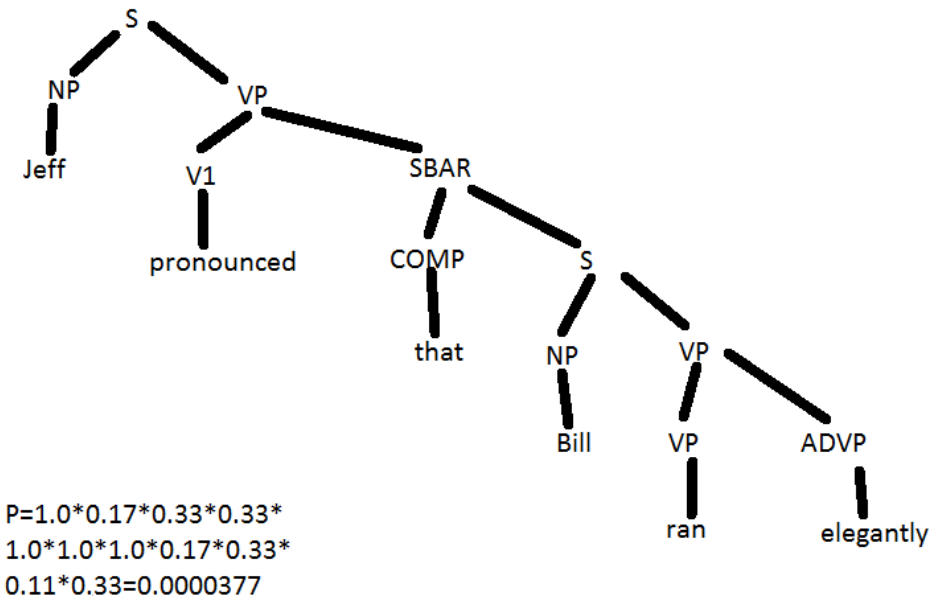
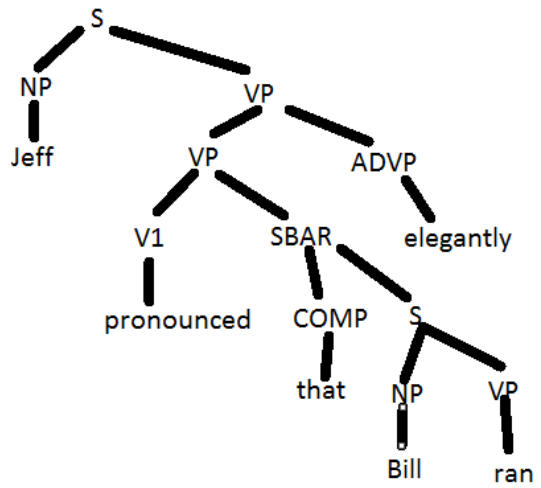


Figure 5: Second Parse Tree



P=1.0*0.17*0.33*0.33*
0.33*1.0*1.0*1.0*0.17*
0.11*0.33=0.0000377

4.2 Grammar Refinement

The possibility of two parse trees arised because of attachment ambiguity where VP can be expanded to V1 and SBAR or VP and ADVP. A tag split on VP can resolve the ambiguity, where VP is divided into VBP and Vt where VBP is the VP which expanded to non-terminals and Vt is the VP that ended in terminals. The modified rules of grammar are:

S → NP VBP [1.0]
VBP → V1 SBAR [0.5]
VBP → Vt ADVP [0.5]
Vt → snored [0.33] | ran [0.33] | swam [0.33]
V1 → said [0.33] | declared [0.33] | pronounced [0.33]
SBAR → COMP S [1.0]
COMP → that [1.0]
ADVP → loudly [0.33] | quickly [0.33] | elegantly [0.33]

Modifying VP this way ensures only one parse tree because for the VP in [1,6] can be expanded in only way.

5 CFG Grammar Refinement II

No. Training the model with more sentences would only make the parser choose a certain structure over the other but inherently the structure is still wrong. Lexicalization would solve the problem because then the model attaches very less probability to the text *"dishes in my pajamas"* and addresses the lexical insensitivity of PCFG which is the main cause of such ambiguities.