Shraddha Agrawal
S.I. 913001594
Winter 2018

Assignment 3: Probabilistic Context Free Grammar

1. This PCFG implements a Bigram model. If we closely look at the grammar 2 i.e.

+Det -> Det 1
+Det -> Det +Det 1
+Det -> Det +Misc 1
+Det -> Det +Noun 1
+Det -> Det +Prep 1
+Det -> Det +Proper 1
+Det -> Det +VerbT 1

The probability of each corresponds to

P(Wn | Wn-1) = count (Wn-1 Wn) / count (total number of Wn-1)
P(Det +Misc | +Det) = 1/7 where 1 is the count(+Det-> Det +Misc) and 7 is count(+Det)

Further, we will prove that the best parse is due to Bigram model using the sentence "Arthur is the king".

"5.973e-12 1.195e-11 0.500 (ROOT (S2 (+Proper (Proper Arthur) (+VerbT (VerbT is) (+Det (Det the) (+Noun (Noun king) (+Misc (Misc .))))))))"

PCFG calculates the probability of each word using Bigram using the following way:

P(Arthur | Proper) * P(Proper +VerbT | +Proper) * P(+Proper | S2) = 1/6* 1/7 * 1/8
P(is | VerbT) * P(VerbT +Det | +VerbT) = 1/7 * 1/6
P(the | Det) * P(Det +Noun | +Det) = 1/7 * 1/9
P(king | Noun) * P(Noun +Misc | +Noun) = 1/7 * 1/21
P(. | Misc) * P(Misc | +Misc) =1/7 * 1/183

And now if we calculate the total probability we get:
P(T,S) = 1/6*(1/7)^5*(1/8)*(1/6)*(1/9)*(1/21)*(1/183) = 5.973*10^(-12).

Which is equal to best parse. Thus, we can assert that the PCFG model utilizes the Bigram language model to give its best parse.

Furthermore, after analyzing grammar 1 and grammar 2 together we can say that grammar 2 is actually backoff model for grammar 1.
As we have

ROOT -> S1 99
ROOT -> S2 1

So, if the sentence fails to parse through the grammar 1 then it backs off to grammar2. The initial grammar S1 and initial backoff grammar 2 are tied together by ROOT symbol.

1.

When we only run the command:

```
./cfgparse.pl grammar1 lexicon < examples.sen
```

We observe that after sentence 2 all the other sentences fail, this is because Grammar 1 does not have any grammar rule explicitly for +misc. And parse trees for all the sentences after 3 are derived from misc and +misc tags, thus grammar 1 fails for them.

But having said that now when we run the command:

```
./cfgparse.pl grammar1 grammar2 lexicon < examples.sen
```

We observe that the best parse for grammar 1 (for the first 2 sentences) is better than the best parse tree for grammar 2 this is because in grammar 2 all the rules have been assigned the weight 1 so the probability of occurrence of any rule remains the same. In actuality, there are some rules that occur more than others, and there are some that are unlikely to be used, so when we assign equal weight to all the rules It reduces the accuracy of the grammar. Whereas grammar 1 assigns different weights to each rule therefore, it is able to predict better.

Now, when we combine both grammar 1 and grammar 2 then its assigns better best parse probability because grammar 1 contributes it weights and grammar 2 contributes more rules, including the rules for misc. So the combine is better than alone grammar 1 or 2.

2.

When I run the three commands these are the results:

```
campus-038-143:hw3 shraddhaagrawal$ ./cfggen.pl --text 10 grammar1 lexicon
1:  any defeater drinks each servant .
2:  this sovereign has the coconut .
3:  any servant is a quest .
4:  any story covers the pound .
5:  each sovereign carries each sovereign .
6:  any fruit carries every quest .
7:  this fruit has any sovereign .
8:  Dingo rides any sovereign .
9:  another servant drinks another swallow .
10:  this pound carries this sovereign .
```

```
campus-038-143:hw3 shraddhaagrawal$ ./cfggen.pl --text 10 grammar2 lexicon
1:  another Zoot Sir Lancelot drinks master every Arthur neither above
2:  rides that each Uther Pendragon defeater chalice
3:  Sir Lancelot drinks
4:  suggests riding were
5:  through through below that each every Dingo defeater into Uther Pendragon Sir Lancelot pound story
any at goes pound Uther Pendragon Guinevere covers having each temperate drink across Dingo
mountains for is fruit one home covers riding that that
6:  quest this Sir Lancelot master land below are through
7:  master Sir Bedevere winter
8:  ridden no is Uther Pendragon on
9:  the quest
10:  above might below trustier for for drinks
```

```
campus-038-143:hw3 shraddhaagrawal$ ./cfggen.pl --text 10 grammar2 grammar1 lexicon
1:  any king covers the chalice .
2:  a home has each horse .
3:  that chalice has the coconut .
4:  this swallow rides the land .
5:  that defeater is this coconut .
6:  every corner covers this land .
7:  the master carries another sun .
8:  another winter covers every quest of the corner .
9:  the weight drinks each swallow .
10:  any fruit covers this castle .
```

These are the key points of differences that I observe:

- Grammar 2 generates the worst sentences because:
  - All the rules have the same weights that is

    S2 -> +Det 1
    S2 -> +Misc 1
    S2 -> +Noun 1
    S2 -> +Prep 1
    S2 -> +Proper 1
    S2 -> +VerbT 1

    So, the grammar does not give preference if the sentence should start with prep, misc or verb. And the sentence starts with anything like punctuations, verbs and nouns which generates incoherent sentences.

- Grammar 1 + Grammar 2 generates better sentence than Grammar 2 because
  - Grammar 1 assigns 99 to ROOT-> S1, thus ROOT would go to S1 99% of the time when grammar 1 and 2 are combined. Whereas in Grammar 2 ROOT -> S2 is assigned 1 so it reduces the probability to go to S1 and thus creates incomplete sentences.
- Grammar 1 generates better sentences in general because as I mentioned in part 2 it assigns different weights to each rules based on the likely occurrence of the rules, which leads to generation of more accurate semantically/ grammatically accurate sentences.
  Mainly grammar 1 is closely following the actual English grammar rules such as S1-> NP VP. Thus it assigns higher probability.

- When we use Grammar 2 it includes rules for Misc due to which there are more additional vocabulary for grammar 1 and 2 combined.

3.

My entropy is coming out to be: 4.05540646579e+17
And the entropy of grammar 1 and grammar 2 is 6.0622846568e+21

These are the following things that I did to improve the grammar:

1. Converted the tags of Misc to Noun, VerbT, Proper, Prep and Det depending on which one the Misc tag is most related to, so this decreases the uncertainty in the grammar rule. i.e. Converted Misc -> speaks 1 to VerbT -> speaks 1 and Misc -> coconut 1 to Noun -> coconut 1
2. Added the EOS tag for '.', '!' and '?' in mylexicon and created new grammar rule for EOS. This makes sure that each sentence generated has an End of Sentence punctuation thus it's they are complete sentences.
3. While going over the sentences, I observed that WHD tags are important i.e. What, Who, When , Where etc. Therefore, I added WHD tag by assigning words to WHD tag in mylexicon of weights 1 . And then in mygrammar adding the rule: S1 -> WHD NP VP EOS 1 as grammatically WHD followed by NP-> VP. This minor changed helped generate complete question sentence.
4. Lastly, I inspected all the mygrammar and mylexicon rules to assign higher weights to more probably rules (that are grammatically correct and occur more frequently in English grammar) and remove the rules that are unlikely to occur, that would increase the accuracy of the best parse.

5. 09/20 lines are grammatically correct. When choosing the grammatically correct I focused on the rules that are grammatically correct rather than actual sense of it. i.e.

a servant knows that master ? // where 'a' doesn't make sense but a Det followed by a noun in a sentence is correct.