Homework project 1: Context Free Grammar

Section One: Generate random sentence with CFG

Q1: generate 10 sentences using CFG:

1.' the delicious chief of staff with a fine president in every pickle ate every pickle in a chief of staff .'
2. ' every floor understood every pickle in a pickled chief of staff in a pickle !'
3. ' a floor in the floor in every president understood every floor !'
4. 'is it true that the pickle ate every sandwich ?'
5. ' a perplexed president under the delicious president ate every chief of staff !'
6. ' the delicious president wanted a president .'
7. ' the perplexed sandwich in the floor in the president under every chief of staff with a pickle under the president on every pickle under the floor on a president under every president on a president with the pickled floor in the president in a pickle on the chief of staff in the floor on a fine sandwich under every fine president on the fine president in a sandwich on every chief of staff on the fine floor in the pickle in every pickle in every president with a sandwich under every chief of staff on every floor in every sandwich with a sandwich on every president with every delicious pickled fine president on the pickled chief of staff on a sandwich in the pickle with every sandwich under every floor in a chief of staff on a president under the pickle with every sandwich under the pickled president on every floor with the pickled fine floor on the pickle with the pickle with the chief of staff under every pickle with every floor in a sandwich under a chief of staff with the floor in every perplexed pickle in every pickled sandwich in every pickle in every chief of staff on every president under a pickle with the sandwich on the chief of staff on every fine president in every chief of staff with a chief of staff on every floor with every president on a fine chief of staff on the sandwich with a ... ... !'
8. ' the sandwich pickled every sandwich .'
9. 'is it true that every sandwich understood the floor with every sandwich with a perplexed pickle in every floor under the pickled fine perplexed sandwich in every pickle with a sandwich on a president on every pickle under a floor on a president on the pickled sandwich on a chief of staff under a chief of staff in a pickle under every pickle with every sandwich on a sandwich in the pickle on the pickle with a president in the sandwich with a pickle in the president on the sandwich on every pickle in the floor under the pickle with a sandwich in every chief of staff with the sandwich with the sandwich ?'
10 'is it true that a president with a delicious chief of staff pickled a floor ?'

Q2: Using string to represent the tree structure.

Structure: (ROOT (S (NP (Det the) (Noun (Adj perplexed) (Noun sandwich))) (VP (Verb kissed) (NP (NP (Det the) (Noun (Adj pickled) (Noun president))) (PP (Prep with) (NP (Det the) (Noun president)))))) .)
Sentence: the perplexed sandwich kissed the pickled president with the president

Structure: (ROOT is it true that (S (NP (NP (Det every) (Noun floor)) (PP (Prep in) (NP (NP (Det the) (Noun (Adj delicious) (Noun sandwich))) (PP (Prep under) (NP (NP (Det every) (Noun sandwich)) (PP (Prep with) (NP (Det every) (Noun (Adj pickled) (Noun pickle))))))))) (VP (Verb pickled) (NP (Det a) (Noun president))))?)
Sentence: 'is it true that every floor in the delicious sandwich under every sandwich with every pickled pickle pickled a president ?'

Q3: Set the max expansion to 5

Structure 1: (ROOT (S (NP (Det every) (Noun floor)) (VP ... ...)) .)
Sentence 1: ' every floor ... ... the floor ... .'

Structure 2: (ROOT (S (NP (NP (NP (Det a) ...) ...) ...) ...) !)
Sentence1: ' a ... ... !'

Section 2: Grammar !

1. Why does your program generate so many long sentences? Specifically, what grammar rule (or rules) is (or are) responsible and why? What is special about it/them?

Answer:

It is quite possible that the CFG tree won't stop expanding till the max_expansions is reached. Of course, in the lucky case, the expansion will be ended very quickly as some nonterminals are converted to preterminal, which generates symbols that can only be expanded to words.

I would say the reason that we have many rules that turn non-terminals to NP explains why we are having so many long sentences. In other words, the NPs and the others keep expanding to the NPs. That is the reason.

2. The grammar allows multiple adjectives, as in the fine perplexed pickle. Why do your program's sentences do this so rarely? (Give a simple mathematical argument.)

Answer:

The relative odds of generating adjective: $\frac{1}{1+sum\ of\ odds\ of\ vocabulary} = \frac{1}{1+5} = \frac{1}{6}$. Therefore it is unlikely for our CFG here to generate adjectives instead of sampling from existing noun vocabularies.

3. Which numbers must you modify to fix the problems in item 1 and item 2, making the sentences shorter and the adjectives more frequent?

Answer:

To solve the above-mentioned issues, we must modify the odds that are assigned to the nonterminals. Specifically, for example, to get more adjectives, one must increase the odds of the rule that converts Noun to Adj and Noun. Increasing the odds of the rule that converts NP to DET and Noun will make the sentence shorter as the nonterminals will quickly be converted to words.

4. What other numeric adjustments can you make to grammar2.gr in order to favor more natural sets of sentences? Experiment. Explain the changes.

Answer:

By observing the tree structure of my CFG, I found that among those of long sentences, they suffer from the issue of overpopulation with NPs. That is, there are many expansions from NP phrases (NP to NP and PP). So in my new grammar file, I will reduce the odds that have been assigned to this rule. I want to make the rule (NP to DET and Noun) more probable.

To make a more natural sentence, I want the CFG to generate short sentences that are as short as possible and as grammatically diverse as possible. In order to meet that end, I decrease the odds associated with NP to NP PP to 0.001 and increase the odds associated with Noun to Adj and Noun to 100 (More grammatically diverse).

The result coincides with my expectation as they are shorter than before and more frantically diverse(multiple adjs appear before a noun).

5. Provide 10 random sentences generated with the grammar2.gr.

Answer:

1) 'is it true that a sandwich wanted the chief of staff ?'
2) ' every chief of staff pickled the chief of staff !'
3) 'is it true that the pickle wanted every pickle ?'
4) ' a floor under the pickle understood every president with a perplexed chief of staff under the president !'
5) ' a pickle wanted the perplexed president under the floor in every chief of staff on the floor on every chief of staff on a president .'
6) 'is it true that the chief of staff understood the chief of staff on every president on a fine floor ?'

7) ' a pickle understood the chief of staff under the president with a sandwich in the floor on a pickle on the floor with **the pickled delicious perplexed president** under the sandwich under the fine sandwich on the floor in the floor on a delicious floor with the sandwich on the floor with the chief of staff under the floor with a pickle in a fine president .'
8) ' every president wanted a floor on the pickle !'
9) 'is it true that a floor ate the chief of staff ?'
10) 'is it true that every president ate every floor ?'

2.3: New grammar file that can generate better natural language.

Major modification that I made to the previous grammar2.gr:

1) I add the rule: NP to Noun to make sentence 1 grammatically possible.
2) I also add the rules VP to V Conj; Conj to vocabularies like AND or OR; Conj to Conj and VP to make sentence 2 grammatically possible
3) Add VP to intransitive_verb to make sentence 3 grammatically possible
4) Add VP to V and Clause; Clause to DE S to make sentence 4 grammatically possible
5) Add S to S and Clause to make 5 possible
6) Add S to Clause and VP to make 6 possible
7) We need adv to describe adj; Add adj to adv and adj to make 7 possible
8) Verb can be converted to Verb and preposition to make the phrase like “work on” possible

Here are the random sentences generation:

1. ' a president wanted every perplexed president on every chief of staff under a sandwich under every chief of staff on the floor !'
2. ' the fine delicious chief of staff understood the floor on a pickled sandwich in the pickle !'
3. ' the chief of staff kissed a president .'
4. 'is it true that every delicious pickle in a perplexed floor in the pickle pickled a pickle ?'
5. ' a pickle ate a chief of staff in the floor in a floor with the floor with the sandwich with every floor with the pickle !'
6. 'is it true that the chief of staff pickled every chief of staff under the fine president in the floor ?'
7. ' the pickled sandwich understood every chief of staff with a delicious floor in a president with the floor under every sandwich on a sandwich under a sandwich under the floor !'
8. 'is it true that every president wanted a perplexed fine pickled floor ?'
9. ' every sandwich ate the pickle .'
10. 'is it true that a floor understood the chief of staff ?'

3.1

a) It is hard to print out the structure by hands; So I will just describe here:
The first the two PPs should not be independent from each other. Rather, the second PP should be generated by the first PP along with the NP.
Now it is like this: **every sandwich (with a pickle on the floor) wanted a president** .

b) Using the first interpretation, one would think the sandwich on the floor wanted a president. But, if we interpret it differently, one might think the pickle is on the floor instead.

3.3

2) Sentence 1: ' the president pickled every fine floor .'
Original : (ROOT (S (NP (Det the) (Noun president)) (VP (Verb pickled) (NP (Det every) (Noun (Adj fine) (Noun floor))))) .)
Parsed: (ROOT (S (NP (Det the) (Noun president)) (VP (Verb pickled) (NP (Det every) (Noun (Adj fine) (Noun floor))))) .) Exactly the same !

Sentence 2: the pickle ate a sandwich !
Original: (ROOT (S (NP (Det the) (Noun pickle)) (VP (Verb ate) (NP (Det a) (Noun sandwich)))) !)
Parsed: (ROOT (S (NP (Det the) (Noun pickle)) (VP (Verb ate) (NP (Det a) (Noun sandwich)))) !) Exactly the same !

Sentence 3: is it true that the chief of staff pickled every sandwich ?
Original: (ROOT is it true that (S (NP (Det the) (Noun chief of staff)) (VP (Verb pickled) (NP (Det every) (Noun sandwich)))) ?)
Parsed: (ROOT is it true that (S (NP (Det the) (Noun chief of staff)) (VP (Verb pickled) (NP (Det every) (Noun sandwich)))) ?) Exactly the same !

Sentence4: the president with every sandwich in the delicious sandwich on every sandwich wanted every chief of staff !
Original: (ROOT (S (NP (NP (NP (NP (Det the) (Noun president)) (PP (Prep with) (NP (Det every) (Noun sandwich)))) (PP (Prep in) (NP (Det the) (Noun (Adj delicious) (Noun sandwich))))) (PP (Prep on) (NP (Det every) (Noun sandwich)))) (VP (Verb wanted) (NP (Det every) (Noun chief of staff)))) !)
Parsed: (ROOT (S (NP (NP (Det the) (Noun president)) (PP (Prep with) (NP (NP (Det every) (Noun sandwich)) (PP (Prep in) (NP (NP (Det the) (Noun (Adj delicious) (Noun sandwich))) (PP (Prep on) (NP (Det every) (Noun sandwich)))))))) (VP (Verb wanted) (NP (Det every) (Noun chief of staff)))) !) Different ! As you can see the difference in the number of parentheses around certain words.

Sentence5: is it true that the chief of staff kissed every pickle in a floor on a pickled chief of staff on a floor with the pickled sandwich in a pickle in a president on a sandwich under the chief of staff in every delicious chief of staff under every sandwich in a delicious sandwich with a pickle under a floor in every sandwich with a chief of staff ?

Original: (ROOT is it true that (S (NP (Det the) (Noun chief of staff)) (VP (Verb kissed) (NP (NP (NP (Det every) (Noun pickle)) (PP (Prep in) (NP (NP (NP (Det a) (Noun floor)) (PP (Prep on) (NP (NP (NP (Det a) (Noun (Adj pickled) (Noun chief of staff))) (PP (Prep on) (NP (NP (Det a) (Noun floor)) (PP (Prep with) (NP (NP (Det the) (Noun (Adj pickled) (Noun sandwich))) (PP (Prep in) (NP (NP (Det a) (Noun pickle)) (PP (Prep in) (NP (Det a) (Noun president)))))))))) (PP (Prep on) (NP (Det a) (Noun sandwich)))))) (PP (Prep under) (NP (NP (Det the) (Noun chief of staff)) (PP (Prep in) (NP (NP (Det every) (Noun (Adj delicious) (Noun chief of staff))) (PP (Prep under) (NP (NP (NP (Det every) (Noun sandwich)) (PP (Prep in) (NP (Det a) (Noun (Adj delicious) (Noun sandwich))))) (PP (Prep with) (NP (NP (NP (Det a) (Noun pickle)) (PP (Prep under) (NP (Det a) (Noun floor)))) (PP (Prep in) (NP (Det every) (Noun sandwich)))))))))))))))) (PP (Prep with) (NP (Det a) (Noun chief of staff))))))) ?)

Parsed: (ROOT is it true that (S (NP (Det the) (Noun chief of staff)) (VP (Verb kissed) (NP (NP (Det every) (Noun pickle)) (PP (Prep in) (NP (NP (NP (NP (NP (NP (NP (NP (NP (Det a) (Noun floor)) (PP (Prep on) (NP (Det a) (Noun (Adj pickled) (Noun chief of staff))))) (PP (Prep on) (NP (NP (NP (Det a) (Noun floor)) (PP (Prep with) (NP (NP (Det the) (Noun (Adj pickled) (Noun sandwich))) (PP (Prep in) (NP (NP (Det a) (Noun pickle)) (PP (Prep in) (NP (Det a) (Noun president)))))))) (PP (Prep on) (NP (NP (NP (Det a) (Noun sandwich)) (PP (Prep under) (NP (Det the) (Noun chief of staff)))) (PP (Prep in) (NP (Det every) (Noun (Adj delicious) (Noun chief of staff))))))))) (PP (Prep under) (NP (Det every) (Noun sandwich)))) (PP (Prep in) (NP (Det a) (Noun (Adj delicious) (Noun sandwich))))) (PP (Prep with) (NP (Det a) (Noun pickle)))) (PP (Prep under) (NP (Det a) (Noun floor)))) (PP (Prep in) (NP (Det every) (Noun sandwich)))) (PP (Prep with) (NP (Det a) (Noun chief of staff)))))))) ?) Different ! As you can see the difference in the number of parentheses around certain words.

As you can see from the above examples, some of the parsed structures are not the same as the original ones. This phenomenon is due to the ambiguity in our natural language. The more complicated the sentence is, the harder it can be for the parser to replicate the tree structure.

3) According to the parser, there are 5 parses to the given sentence.

(NP (NP (NP (Det every) (Noun sandwich)) (PP (Prep with) (NP (Det a) (Noun pickle)))) (PP (Prep on) (NP (NP (Det the) (Noun floor)) (PP (Prep under) (NP (Det the) (Noun chief of staff))))))

4) Generate sentences using grammar.gr

every pickle in the president in a chief of staff with a chief of staff in every president with every pickle with every pickle kissed every sandwich in the floor with a floor !

 # number of parses = 264

the fine pickle in a chief of staff under the pickle with a chief of staff under the chief of staff under every pickle understood every chief of staff .

 # number of parses = 42

a floor on the chief of staff in every pickle with every delicious delicious sandwich under a president under every president with every chief of staff with every president on a floor on a sandwich in a president with a floor with the president with the pickled pickled chief of staff in the chief of staff in a perplexed floor on the floor with every sandwich with the delicious pickle with the floor under every pickle on a floor under a chief of staff under a president on every president in the sandwich understood the pickle .
 # number of parses = 4861946401452

Generated using Grammar 3:

is it true that the president wanted a chief of staff under the fine perplexed fine chief of staff under a pickle in the sandwich in the sandwich on the sandwich with a sandwich ?
 # number of parses = 132

is it true that every pickled chief of staff under the floor under the chief of staff under every sandwich with every floor on a chief of staff on a president ate every delicious floor in every chief of staff ?
 # number of parses = 132

the chief of staff on the sandwich in the sandwich pickled a president **.**
 # number of parses = 2

I think the pattern is that a complicated sentence usually has lots of parses. A better written grammar with reasonable odds assigned to the rules also has less parses in general.

5)

(a) P(best_parse) is a subset of the probability of the generation of sentences. Since there are only one way of parsing in this case, P(best_parse)=p(sentence)=(½*½*⅓*⅙*⅕*⅓*⅙*⅓)= 5.144e-05
P(best_parse | sentence)= P(best_parse and sentence) / p(sentence)
We know that P(best_parse and sentence)=P(best_parse)=p(sentence) as there is only one way of parsing the sentence. This leads to P(best_parse | sentence)=1

(b) Recall that P(best_parse | sentence)= P(best_parse and sentence) / p(sentence)
And P(best_parse and sentence)=P(best_parse) since best_parse is a subset of probability of generating this sentence.
An intuitive explanation is that among these many ways of generating such sentences, best_parse, according to our parser, is the most probable way of generating the sentence

(c) The cross entropy is the average of negative log probability of a word being generated. Therefore, cross-entropy = -log(p(sentence 1))-log(p(sentence 2))= 43.833, where the P(sentence) is already provided in the sentence.

(d) 2^{2.435}=5.40764

(e) Notice that the second sentence can not be generated from the given grammar. Thus P(sentence)=0, which will give a cross entropy of infinity.

6)

a) **python randsent.py -g grammar2.gr -n 1 | ./parse -g grammar2.gr -P** This command will give you cross entropy directly.

Grammar2 can generate the sentence 4.965 bit per word of cross entropy.

b) I run the experiments multiple times and found that the entropy of grammar2 is usually less than that of grammar3.
c) The entropy of grammar 1 is lower than that of 2 and 3. However, it is unstable as it can generate many sentences that are not yet finished. So it will yield corpus that can not be used for calculating entropy.

7) Here are the codes:
python randsent.py -g grammar.gr -n 1 | ./parse -g grammar2.gr -P
4.695
python randsent.py -g grammar2.gr -n 1 | ./parse -g grammar2.gr -P
3.226
python randsent.py -g grammar3.gr -n 1 | ./parse -g grammar2.gr -P
4.899

Section 4: New grammar rules:

a) An vs A
I will handle this problem via grouping the noun and adj, and expand the rules for the expansion of NP.
For example, there will be two groups of determiners. One is for the normals and the other is for the special “An”. We can also do the same for Adj and Noun whose starting letter is a-e-i-o-u
Group1: for the special cases:

```
1   Det1 an
```

```
1   Adj1 ambivalent
1   Noun1   apple
```

Group2: normal cases

```
1   Det    the
1  Det    a
1  Det    every
```

```
1   Noun   president
1  Noun   sandwich
1  Noun   pickle
1  Noun   chief of staff
1  Noun   floor
```

```
1   Adj    fine
1  Adj    delicious
1  Adj    perplexed
1  Adj    pickled
```

Add these rules which allow the generation of an

```
1   NP  Det1 Noun1
1   Noun1   Adj1 Noun
1   Noun1   Adj1 Noun1
```

b) Yes and no question

We just need to expand the rules for ROOT.

LIke in this case:

```
1   ROOT    did S ?
1   ROOT    will S ?
```