Dept. of Computer Science, University of California, Davis
ECS189G Weiran Guo(912916431) March 6, 2018

HW3

1 Task 1

This PCFG uses HMM model, that generates all the possible POS tagging of the words, the best parse is the most likely sequence of POS tagging.

$$P(sentence) = \frac{1}{6} * \frac{1}{7}^{NumOfTerminals} * \prod_i p(terminal_i \ in \ certain \ tag \ set) \tag{1.1}$$

which $\frac{1}{6} * \frac{1}{7}^{NumOfTerminals}$ is considered a constant for a certain sentence, the P only depends on the probability of the word in a certain tag set.

Note that if P(T|S) = 0.5, there is only one parse for the sentence. This means for each terminal in lexicon, there is only one POS tag for it.

2 Task 2

For only using grammar 1, it simply cannot parse most of the sentences, which means there is no grammar rules can satisfy the formation of the sentence.

If using both grammar 1 and grammar 2, it can parse all the sentences. Also, notice that for all the sentences that grammar 1 cannot parse, the merging grammar generates exactly the same best parse tree as only using the grammar 2. After skimming through the grammars, I found that it it because the weight of S1(99 in grammar 1) is much bigger than the S2(1 in grammar 2), so for sentences that grammar 1 can parse, the best parse will always be generated by grammar 1, else it will be generated by grammar 2.

3 Task 3

By setting N to 10, I found that all the sentences that grammar 1 generates are grammatically correct, however, none of sentence that grammar 2 generates are correct! By

merging grammar 1 and 2, I found all of them are grammatically correct.

As task 1 shows, the grammar 2 only deals with POS tagging with HMM model, which does not involve grammar rules, hence that is why sentences generated by it are not grammatically correct.
On the other hand, the grammar 1 focuses on the grammar rules, hence it generates all the sentences grammatically correct, though in the Task 2 it failed to parse many real sentences that does not fit the rule.
The reason that merging rules generate correct sentences, is that grammar 1 weights much more than grammar 2 in the merging grammar. As Task 2 discovered, the S1 weights 99 while S2 only weights 1. This explains why the rules including grammar 2 still generates mostly correct sentences.

4 Task 4

For the merged grammar and provided lexicon, I got the cross entropy score of:

```
MacBook-Pro-2:hw3 weiran$ ./cfgparse.pl grammar1 grammar2
    lexicon < examples.sen > temp
MacBook-Pro-2:hw3 weiran$ python cross_entropy.py
Cross entropy =  72.36035158862143
```

By using my grammar and lexicon, I got

```
MacBook-Pro-2:hw3 weiran$ ./cfgparse.pl mygrammar mylexicon <
    examples.sen > temp
MacBook-Pro-2:hw3 weiran$ python cross_entropy.py
Cross entropy =  69.22733710834038
```

I have several approaches.
1. assign new types(Adj, Pronoun, Part) for words in kind Misc, as task 1 shown, it will increase $p(terminal_i \textit{ in certain tag set})$, hence increase the sentence probability.
2. Add new rules in S1. This will let more sentences be parsed by S1 instead of S2, hence increase the probability of the sentence.
3. Add Rules in S2, this will ensure it will not fail to parse the sentences.
4. lower the weight of S1 a little bit, hence increase the cross-entropy while not decreasing the grammatical accuracy of sentences generated by the rule.

5 Task 5

18 out of 20 sentences generated by the grammar are grammatical.