497 A2 Report

Marc-Andre Haley • Chu Duc Thang

Results

**** NOTE: We interpreted the data in this manner:*

Ground Truth

Prediction

	0	1
0	True Positive	False Positive
1	False Negative	True Negative

Our final grammar gave us the following results:

- True Positives: 286
- True Negative: 52
- False Positives: 87
- False Negatives: 178

From a total of 603 sentences.

Precision

Precision = TP/(TP + FP) = 286/(286 + 87) ≃ 0.77 or 77%

Recall

Recall = TP/(TP + FN) = 286/(286 + 178) ≃ 0.62 or 62%

Error Analysis

Overall, our Grammar performed as expected. Not particularly well but not bad either. We started with the toy grammar and added some common rules mentioned in chapter 12 of the book. Then we went through some of the data sentences and added rules accordingly. At first we had a lot of False negatives because no sentences were parsed with the rudimentary grammar. As we went through the sentences and edited the rules accordingly we started getting a lot less false negatives and more true positives. However, as the true positives increased, the false positives were increasing as well because some of the rules we were adding were not perfect and would generalize to accept sentences with errors as well. Reasons for errors:

- **False Positives:** As mentioned above, as we added rules, the grammar would start to be too broad in some areas which allowed the parser to parse some sentences with grammatical errors. Another reason for some false positives is that some sentences might have been wrongly labeled as erroneous. For example “Yes, it was strange behaviour.” and “That change really disappointed me.” don’t seem to be grammatically incorrect. Or some sentences had typos in them which resulted in them getting labeled as erroneous. For example, “The entery will be free.” was labeled with 1 because of the typo, but had pos “DT NN MD VB JJ .” which is a valid sentence.

- **False Negatives:** The main reason for the false negatives is that we simply didn’t go through all the sentences so there weren't enough rules to correctly parse every sentence. There were also some pos tags that we did not know how to write rules for. For example, tags like -LBR-/-RBR- and HYPH were hard to implement in the rules because they could almost go anywhere. Since these tags didn’t get included in our grammar, any sentence with these tags could not be parsed, even if it might be correct.

Report questions

- *With our current design, is it possible to build a perfect grammar checker?*
  Answer: It would be possible to build a perfect grammar checker for the given finite set of sentences in input/train.tsv. We would simply need to go through every sentence in the dataset and edit the grammar rules so they give a correct prediction for that sentence. However, tuning a rule set this way would likely not generalize to all of english. So it’s probably not possible to build a perfect grammar checker for English, as I will prove below.
- *If not, briefly justify your answer.*
  Answer: Building a perfect grammar checker for English this way would require us to write a perfect CFG for English. If we built a perfect CFG for English then we would prove that English is context-free. Since English is probably NOT context-free(*), then we cannot build a perfect grammar checker.
  (*) https://www.jstor.org/stable/4178381