CMPUT 501: Intro to NLP

Assignment 2 - Grammar Checker

By: Mohammad Karimiabdolmaleki & Ali Zamani

Classification Report

		Ground Truth	
		1	0
Predicted	1	56	140
	0	83	324

Precision = TP / (TP + FP) = 0.29
Recall = TP / (TP + FN) = 0.40

- **How and why the grammar checker produced false positives and negatives?**
  As shown above, false positives occur when the predicted value by the grammar checker is 1 while the ground truth that we have from the train.tsv file is equal to 0. On the other hand, false negatives occur when the predicted value by the grammar checker is 0 while the ground truth is equal to 1. That being said, the CFG rules that we came up with, were not able to handle all of the grammatically correct sentences that were present in the train.tsv file and that caused the false positives. Also, in some cases, the CFG rules that we designed, parsed the sentences that should not be parsed, since the true label was 1 which means that the sentence is not parsable. This was the reason that we had false negatives in our results. Unfortunately, in this problem, we can not have a very high value for both of the values and it is well-known as Precision-Recall-Tradeoff, meaning that increasing one of them will result in decreasing the other one. During our progress with the CFG rules, we have seen so many different values for precision and recall and we realized that they are negatively correlated.

- **With our current design, is it possible to build a perfect grammar checker? If so, what resources or improvements are needed? If not, briefly justify your answer.**
  No - We do not believe that with the current design, we are able to build a perfect grammar checker. There are several reasons behind this claim.

➢ The first problem is that the manual approach is totally obsolete and out-dated (specially in the era of deep neural networks). Looking at each of the sentences and writing a rule and trying to generalize the similar rules is a task which takes lots of effort and time and probably will not have the desired results at the end of the day. Specifically speaking, one may insert more CFG rules with the goal of parsing more grammatically correct sentences, but it turns out that you have also parsed some grammatically incorrect sentences that should have been ignored by the parser.
➢ With every new CFG rule, we are indeed increasing the runtime of the program exponentially. Today that we can handle much more complicated problems with machine learning, parsing a sentence with a toy grammar should be also doable within seconds. That being said, computation cost and complexity of solving the parsing problem with the given instructions is very high. Long runtime and high complexity is not a property of a perfect grammar checker, as mentioned in the question above.
➢ Considering Grammarly[1] as a perfect grammar checker program that can detect grammatical errors within seconds, the approach that they are using are machine learning, deep learning and probabilistic language models with Natural Language Processing techniques that allows the program to identify whether it is grammatically correct or not. Consequently, it can be concluded that for having a perfect grammar checker, we need better approaches than trial and error.

Error Analysis:
To investigate true positive and false negative cases more thoughtfully, we logged all of them in a file (error.txt) in order to make useful inferences. In the following, some cases are analysed and mentioned:

False Positives:
- RB , IN NN . : Based on the English grammar and the course lectures, a full grammatically correct sentence is a sentence that has a noun phrase and a verb phrase (although we have sentences that only contain a verb phrase, but their frequency is much less than the other group) . Consequently, we ignored the

sentences that do not follow this rule. “RB , IN NN .” does not have a verb phrase, hence, we did not place a rule to cover it.

- PRP VBP PRP VBZ JJ VBG PRP . : For some cases, if we add a new rule we can cover that specific case, but we will cover other cases that we should not cover. So, it is better to avoid a rule for such sentences that has a low frequency across the corpus. For instance, we wrote a rule that covered “PRP VBP PRP VBZ JJ VBG PRP .”. However, we figured out that other cases were parsed that should not be parsed.

False Negative:

- PRP VBP IN JJ NN . : In our CFG rules, there is a rule that parses this sentence, and if we remove that, the parser won’t parse this sentence anymore. But, the point is that the parser will not parse other sentences which should parse, either. Hence, there is a tradeoff for adding new rules to the grammar. For some cases like this, we ended up having a rule since it improved false positive rate more than false negative.

Resources:

1- https://www.grammarly.com/blog/engineering/grammarly-nlp-building-future-communication/