Daniel Knafel
316012624

Yanir Buznah
207631466

Natural Language Processing – Assignment 2

1. **Why does the program generate so many long sentences? Specifically, what grammar rule is responsible or that and why? What is special about this rule? discuss.**
   The reason is the relatively high weights (all equal) assigned with **rules that generate non-terminals**, mainly $NP \rightarrow NP\ PP$ which is also recursive. This property of the grammar allows it to recursively generate more and more non-terminals, as well as other parts of speech along them, making the sentences longer.

2. **The grammar allows multiple adjectives, as in: "the fine perplexed pickle". Why do the generated sentences show this so rarely? discuss.**
   First, we noticed that the only way to generate an 'Adj' symbol is from a 'Noun' symbol by using the $Noun \rightarrow Adj\ Noun$ rule. Second, by looking at the weights associated with 'Noun', we noticed that a 'Noun' symbol generates a terminal with probability $\frac{5}{6}$ (there are 5 rules), while it generates an 'Adj Noun' with probability $\frac{1}{6}$ (there is only one rule). This causes adjectives to appear 5 times less frequently than nouns.

3. **Which numbers must you modify to fix the problems in (1) and (2), making the sentences shorter and the adjectives more frequent? Verify your answer by generating from the grammar. Discuss your solution (which rules did you modify, and why).**
   To fix issue (1), we raised the weighs of rules that generate pre-terminals over rules that generate non-terminals, thus effectively lowering the weights of the recursive rules and terminating the sentences earlier.

   To fix issue (2), we increased the weight of $Noun \rightarrow Adj\ Noun$ over the $Noun \rightarrow * terminal *$ rules. To avoid overcompensation and generating too many adjectives using this rule, we needed to balance these weights to about $1:3$.

4. **What other numeric adjustments can you make to the grammar in order to favor a set of more natural sentences? Experiment and discuss.**
   To get more 'natural looking' sentences, we implemented the following changes:
   - Increasing the weights of $ROOT \rightarrow S$. over the other ROOT rules. The reason behind this change is that the vast majority of sentences in English are not questions or exclamatory sentences.
   - Balancing the weights of less commonly used words, such as 'pickled' and 'perplexed', in comparison to words like 'ate' and 'fine'.
5. **Furthermore, note that handling sentences (b) and (h)/(i) can interact in a bad way, to create ungrammatical sentences. You do not need to solve this issue in this part of the assignment, but you do need to discuss it and explain what the problem is, using an example and a short explanation.**
   - The problems we faced was that the word 'is' is a verb, and to generate sentence (b) one could use the naïve rule $VP \rightarrow Verb\ Conj\ Verb$, making sentences like "Sally and the president wanted and is a sandwich", or "Sally and the president are and is a sandwich".

- Furthermore, to generate sentence (i), one could use the rule $VP \rightarrow Verb\ Verb\ NP$, which can interact with (b) to make sentences like “Sally and the president wanted ate and ate a sandwich .”.

6. Briefly discuss your modifications to the grammar (Part 2).

- We separated the Verbs into subcategorizes, based on the arguments they can take. For example, the word ‘sighed’ is V_0 takes no arguments (“the president sighed”), while the word ‘ate’ is a V_1 and takes 1 argument (“Sally ate a sandwich”). Note that some words can be both V_0 and V_1 (‘kissed’).
- We introduced the RB symbol to differentiate between adjectives (“lazy”) and adverbs (“very”) to make sentence (f).
- To make sentence (d), we further separated verbs that can be followed by “that…” into a different symbol, ‘Vthat’, and used the rule $VP \rightarrow Vthat\ that\ S$.
- We added the rule $NP \rightarrow NP\ CC\ NP$, when CC is a conjunction.
- In contrast to the previous rule, $VP \rightarrow VP\ Conj\ VP$ is a bad idea. Instead, we introduced $VP \rightarrow V_1\ Conj\ V_1$, in order to enforce the arguments for the verbs.
- Also, we use other rules such as $NP \rightarrow NNP$, $NP \rightarrow Pron$.

7. Briefly discuss your solutions and provide sample output (Part 4).

We chose to implement sections (a) and (f).

- <u>‘a’ vs ‘an’:</u>
  To enforce this, we added the rule $NP \rightarrow Single\_NP$. A “Single_NP” would generate the appropriate prefix (‘a’ or ‘an’) together with a VOWEL or a CONSONANT, respectively. Those would further generate nouns, adj+nouns or RB+adj+nouns in such a way that ensures an appropriate first word, to match the prefix.
  For example, the phrase “an ambivalent apple” would be generated by:
  $$NP \Longrightarrow \text{Single\_NP} \Longrightarrow \text{an VOWEL} \Longrightarrow \text{an ADJ\_VOWEL Noun} \Longrightarrow \text{an ambivalent apple}$$
  While the phrase “a very ambivalent apple” would be generated by:
  $$NP \Longrightarrow \text{Single\_NP} \Longrightarrow \text{a CONSONANT} \Longrightarrow \text{a RB\_CONSONANT Adj Noun} \Longrightarrow \text{a very ambivalent apple}$$
- <u>Tense and Aspect:</u>
  We continued using the V_0, V_1 approach but added the 4 unique forms of a verb (present simple, present progressive, present perfect and future simple).
  We then added rules to accommodate the correct auxiliary (will, has, has been) along with the matching verb form.
  For example, to generate the phrase “the president has been eating a sandwich":
  $$\begin{aligned} S &\Longrightarrow NP\ VP \Longrightarrow the\ president\ VP \Longrightarrow the\ president\ V_1\ NP \\ &\Longrightarrow the\ president\ V_1\_PAST\_TENSE \\ &\Longrightarrow the\ president\ had\ V_1\_PERFECT\_PROGRESSIVE\ NP \\ &\Longrightarrow the\ president\ had\ been\ V_1\_PROGRESSIVE\ NP \\ &\Longrightarrow the\ president\ had\ been\ eating\ NP \\ &\Longrightarrow the\ president\ had\ been\ eating\ a\ sadwich \end{aligned}$$

8. Provide a discussion of your changes (part 5).

- Subordinating clauses – Chaining 2 clauses using SBARs such *as if, because, unless* etc. the main rule in this section is $ROOT \rightarrow S\ SBAR\ S$.
  We needed to separate them from regular conjunctions (and, or) because they are only interchangeable in one way. Consider the following:

Conj	Change direction	SBAR	Good/Bad
Sally **and** the president	$\Rightarrow$	Sally **if** the president	**Bad**
Sally has eaten **and** the president pickled	$\Leftarrow$	Sally has eaten **because** the president pickled	**Good**

  We wanted to allow even more branching, in the form of $S \rightarrow S\ SBAR\ S$, but found it to be too recursive.
- Questions – generating simple *where is, what is, how much* and *how many* questions.
  $$S? \rightarrow Where\ is\ NP?$$
- Q&A – We implemented a mechanism for enforcing that the answer would contain **the same** noun as the question by using wild-card non-terminals. We used the following rules:
  $$QUTOE\_ENTITY^* \rightarrow NNP \mid Det\ Noun$$
  $$QA \rightarrow What\ is\ QUTOE\_ENTITY^*?\ QUTOE\_ENTITY^*\ is\ NP$$
  for example:

where is	what is	how much	how many
Where is Sally? Sally is on the floor.	What is every pickle? Every pickle is Rick.	How much salt? 2 kilograms.	How many oranges? 2 oranges.

  - We added a category of PPs to answer *where* questions – *on, in, by, at* etc.
  - To address the difference between *much* and *many* we separated our nouns into count nouns and non-count nouns, to be used along with a count (5 apples, 2 people) or mass/volume (5 liters of water, 2 cups of flour).
- Quote phrases - We added three kinds of quote phrases – regular (ending with a dot), questions and exclamatory. Each kind of phrase has a unique set of verbs. For example:

	Regular	Questions	Exclamatory
Example	"Sally has been pickling", Harry said.	"Where is the president?", Joe asked.	"The president kissed the desk!", the apple cried.
More verbs	Responded	Wondered, questioned	Announced, declared, yelled