NLP course Assignment 3: Grammar Engineering

Daniel Bazar 314708181

Lior Krengel 315850594

Part 1 | Weights

1. **Why does the program generate so many long sentences? Specifically, what grammar rule is responsible for that and why? What is special about this rule? discuss.**
   The program generates so many long sentences mainly because of some recursive grammar rules.
   Specifically, the rule: NP → NP PP is responsible for that phenomenon. Since every rule in this part has the same weight, and we start from the rule S → NP VP, then, we have 50% chance of choosing the above rule (because there are two rules for NP). In addition, the VP rule is also derived to a rule that includes NP. Moreover, the nonterminal PP that is derived from the NP rule contains in itself the nonterminal NP which means that NP is called at least twice.

2. **The grammar allows multiple adjectives, as in: "the fine perplexed pickle". Why do the generated sentences show this so rarely? discuss.**
   Generated sentences show multiple adjectives so rarely because in order to get such an event we need to derive the rule Noun → Adj Noun twice. This rule is one of 6 rules with Noun in the LHS, and all rules have the same weight, so the probability for it is just 1/6.

3. **The grammar format allows specifying different weights for different rules. 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).**
   First, to make sentences shorter we need to handle the problem with the NP → NP PP rule we described in (1). The problem was that this rule is chosen too often. so we increased the weight of the second NP rule (NP → Det Noun) to 5 making the relative ratio between the two rules in favor of the "non-recursive" one.
   Second, to make multiple adjectives more frequent, we took the same approach, and we increased the weight of the Noun → Adj Noun rule to 5 to make it more frequent.
4. **What other numeric adjustments can you make to the grammar in order to favor a set of more natural sentences? Experiment and discuss.**
   In order to make sentences more natural, we have adjusted the weights of the rules. This adjustment reflects the likelihood of word distribution. To estimate this, we have adjusted the weights of verbs and adjectives based on the nouns they are commonly paired with. For each verb or adjective, we have assigned a weight based on how many nouns can appear with it in the same context. For instance, for the verb 'ate', we assigned the weight 2 because it can connect with two nouns out of five (one can 'ate' a sandwich and a pickle, but not a president, a chief of staff, or a floor). Similarly, we determined the weights of the nouns based on the verbs and adjectives they are commonly paired with. We didn't change the weights of the

determiners and prepositions because they are likely uniformly distributed.
We have also adjusted the weights of the rules used to create full sentences. We increased the weight of a sentence ending with a period (".") because it is more likely than sentences ending with an exclamation mark ("!") or a question mark ("?").

Part 2 | Extending the Grammar

We made modifications to the grammar in order to generate the types of phenomena illustrated in the given sentences. All explanations can also be seen in the “grammar2” file.

In order to solve this part problem, first we tagged each word in each sentence with the respective part-of-speech. Then we built a syntax tree and from it, we derived the below rules.

For each sentence, we provide the corresponding modifications:

(a)

i. NP → Nnp – to support proper nouns (people, locations, organization, etc.)
ii. Nnp → Sally – adding “Sally” to vocabulary as Nnp

(b)

i. NP → NP Cc NP – to support coordinating conjunction (“…and…”)
ii. VP → Verb Cc VP – to support conjunction between verbs (“…and…”)
iii. Cc → and – adding “and” to vocabulary as Cc

(c)

To respect the *subcategorization frame of verbs* we reclassified the ‘Verb’ preterminal as transitive verbs (V1) and we added another preterminal called ‘V0’ as intransitive verbs.

i. VP → V0 – to support intransitive verb phrase
ii. V0 → sighed – adding “sighed” to vocabulary as V0

(d)

i. VP → Verb Sc S – to support subordinating conjunction (“<dependent clause> *that* <independent clause>”)
ii. Verb → thought – adding “thought” to vocabulary as Verb
iii. Sc → that – adding “that” to vocabulary as Sc

(e)

i. NP → Prp – to support personal pronoun (substitute to person like I, you, and *it*)
ii. NP → NP Sc S – to support subordinating conjunction with NP
iii. Verb → perplexed – adding “perplexed” to vocabulary as Verb. It’s Important to note that “perplexed” has already been used but in different meaning as Adj.
iv. Prp → it – adding “it” to vocabulary as Prp (personal pronoun as described above)

(f) + (h)

After reaching sentence (h) we realized a strong connection between the two sentences, so we made some adjustments. Therefore, the final rules are derived from the sentences together.

i. NP → Det ADJP Noun – to support adjective phrase.
ii. ADVP → Rb – to support adverb phrase
iii. ADVP → Rb ADVP – to support recursive adverb phrase (“very very…”)
iv. Rb → very – adding “very” to vocabulary as Rb (adverb)
v. VP → Vbz ADJP – to support 3rd person singular present (“is…”)

vi. ADJP → Adj – to support adjective phrase
vii. ADJP → ADVP Adj – to support modifying adjectives
viii. Adj → lazy – adding “lazy” to vocabulary as Adj
ix. Vbz → is – adding “is” to vocabulary as Vbz (new preterminal, Verb, 3rd person singular present)

(g)

i. VP → Verb PP – to support PP (providing additional information) also to Verb
ii. Verb → worked – adding “worked” to vocabulary as Verb
iii. Noun → proposal – adding “proposal” to vocabulary as Noun
iv. Noun → desk – adding “desk” to vocabulary as Noun

(h)

Described above at (f)

(i)

i. VP → Vbz Vbg NP – to support a form of present progressive
ii. Vbg → eating – adding “eating” to vocabulary as Vbg (Verb, gerund or present participle. Ending with -*ing*)

(j)

Important note: we are asked in this part to generate *all* sentences given. In this, the word “sally” is written with a lowercase “s”. Despite this, we referred to it like the Nnp word “Sally” and we didn’t make another terminal “sally”.

i. VP → Vbz NP – to support present simple

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.

In sentence (b) we created rules for conjunction between phrases and in sentences (h)/(i) we created rules for linking (“is”). The potential grammatical issue arises when attempting to combine sentence structures (b) and (h)/(i). example for ungrammatical sentence: "Sally and the president is lazy."
The problem is when coordinating "Sally and the president" it’s a plural subject. However, combining the linking “is”, leading to a grammatical disagreement in number. The correct form would be "Sally and the president are lazy".

Part 3 | Tree Structures

We added another optional command line switch -t that generates the sentence tree structure.

All modifications can be seen in the code *generate.py*. specifically, we mainly modified the function *gen*.

Part 4 | Additional Linguistic Structures

The two linguistic phenomena we chose are **Relative clauses (c) and WH-word questions (d).**

(c) Relative clauses

A relative clause, typically modifying a noun or noun phrase, is often introduced by a relative pronoun (such as which, that, who, etc.). It connects ideas by using pronouns referencing previously mentioned elements, enabling the writer to unify two independent clauses into a single sentence. There are two kinds of relative clause: to make clear which person or thing we are talking about and to give more information about a person, thing or situation.
To handle this, we first had to little bit modify our grammar. The relative pronoun that we used are which, that and who. But “which” can be only attached to a “thing” and “who” can only attached to “people” (“that” can be attached to both). So, to handle it we changed the Noun preterminal to a new nonterminal NN that can derive two nouns, Nount – “things nouns” and Nounp – “people nouns”. For example, we changed “president” to Nounp and “sandwich” to Nount. Now, to derive relative clause we used the following rules:

NP → Det Nount RELCT	To make clear or give more information on Nount
NP → Det Nounp RELCP	To make clear or give more information on Nounp
NP → Nnp RELCP	Relative clause for proper noun
RELCP → Relpp VP	Relative clause for people. typically indicating an action associated with the noun
RELCP → Relpp S	Relative clause for people. typically indicating a state associated with the noun
RELCT → Relpt VP	Relative clause for things. typically indicating an action associated with the noun
RELCT → Relpt S	Relative clause for things. typically indicating a state associated with the noun
Relpp → that	Relative pronoun for people
Relpp → who	Relative pronoun for people
Relpt → that	Relative pronoun for things
Relpt → which	Relative pronoun for things

Examples:
“a chief of staff that worked in a sandwich is delicious !”
“a president who sighed ate !”
“the chief of staff that is very fine ate the floor that kissed a president that kissed in a pickle .”
“every sandwich which wanted every chief of staff sighed .”
“Sally who is very fine perplexed a floor !”
“every desk wanted the chief of staff that is the president !”

(d) <u>WH-word questions</u>

We didn't do (b) so we made our life easier by handling "I wonder" sentences with so-called "embedded questions". Embedded questions are part of a statement (in our case "I wonder"). Also unlike regular questions, embedded questions ends with period instead of question mark, don't invert word order and also don't include Auxiliary verb what actually makes our life easier. To handle this, we first "designed" structures of question that can follow the "I wonder" intro and decided which WH-word fit to each one as described in the table below:

	S	**VP**	**S Prep**	**Nount S**	**Nount S Prep**	**Adj NP is**	**NP is**
what	V	V	V	V	V		V
where	V		V				V
when	V						
who		V	V				V
which				V	V		
why	V						
how	V					V	

Then we transformed it to the following rules:

1. ROOT → I wonder EQ .
2. EQ → Ws S
3. EQ → Wvp VP
4. EQ → Wsp S Prep
5. EQ → Wns Nount S
6. EQ → Wns Nount S Prep
7. EQ → Wadj Adj NP is
8. EQ→ Wsp NP is
9. Ws→ what | where | when | why | how
10. Wvp → what | who
11. Wsp → what | where | who
12. Wns → what | which
13. Wadj → how

Examples:

"I wonder where the sandwich is eating the president ."

"I wonder who is the chief of staff ."

"I wonder who the chief of staff is fine with ."

"I wonder what sandwich a fine chief of staff ate with every president."

"I wonder which desk a floor is fine with ."

"I wonder how pickled a pickle is ."

"I wonder who the president is ."

Part 5 | Extra

(a) Post-processing

To let the parts of the sentence coordinate more closely you may add features to non-terminals, this means to duplicate rules and distinguish them by adding suffix to the rule for example. However, it's hard to handle everything with tons of attributes on the nonterminal. We came up with a solution to compose the PCFG with a post-processor. First, we generate a basic sentence then we run it through a post processor to clean it up for external presentation. Some examples of usage we will show here are capitalization, handling unnecessary symbols such as commas and spaces (it's mostly for presentation but it can also help us in some cases as we will see for appositives), changing "a" to "an" before a vowel, and some beautifications. All extensions are in *generatex.py*.

Capitalization:
Our basic sentences mostly starts lowercased so we capitalize the sentences.
Examples:
"The officer kissed a very pickled president ."
"A president that wanted a very very very fine chief of staff ate on it ."

handling unnecessary symbols such as commas and spaces (**appositives**):
When working on part 4 we experimented with phenomena (g), appositives. We struggled with it. The tricky parts are (a) handling multiple commas (b) ending each appositive with a comma unless it comes at the end of a sentence. We actually found a way to handle each problem, but the problem was that one came at the expense of the other. There were also some more minor problems to address like preventing "that" from following a comma. Eventually, we did (d) WH questions and abandoned appositives. But when the idea to post-process came up, we also realized that we could implement this more easily. So, we already had the rules for creating appositives but with the limitations of duplicates and trailing commas, so we added those rules to the grammar and a step in the post-process to handle it. The rules:

1. NP → Det NN Comma APP Comma
2. NP → Det ADJP NN Comma APP Comma
3. NP → Nnp Comma APP Comma
4. APP → APP Comma APP
5. APP → YEARSOLD
6. APP → Nnp
7. APP → RELCP
8. APP → RELCT
9. Comma → ,
10. YEARSOLD → Num years old
11. Num → 25 | 59 | 18 | 72

Examples:
"I wonder what sandwich Sally , 18 years old , ate it with ."
"Is it true that a chief of staff , 59 years old , is fine ?"

“a” vs. “an”:

We add some more vocabulary words ends with vowels and changed “a” to “an” if its before vowel. The new words:

Nounp → ambassador | officer | apple | umbrella | ice cream

Adj → elegant | angry

Examples:

“An officer sighed .”

“A perplexed officer , Sally , is an umbrella . ”

Beautification:

Another step in our post-process is improving the visibility of the sentence and making it to appear more natural without unnecessary whitespace and such.

For example: removing leading and trailing whitespaces (before and after “.”, “?” , “!”, “:” or ‘”’), removing whitespaces before commas and removing whitespace before -s’ suffix (possessive marker).

Examples:

“An officer, Sally, 18 years old, is an umbrella.”

“Is it true that Sally, 72 years old, who ate, understood an elegant officer, 72 years old?”

(b)Quotes

We added to our grammar rule for creating quotes of the form:

ROOT → " S . " NP said .

Examples:

‘"A delicious umbrella, who is angry, is an apple." Sally, 59 years old, said.’

‘"An officer is eating a sandwich." a fine ambassador said.’

(c)Ditransitive verbs (V2)

Ditransitive verbs are used when someone other receives something as the result of the action of the verb.

We added the following rules and terminals to handle ditransitive verbs:

VP → Vd NP NP

Vd → gave | sent | showed

Examples:

‘"Sally, 25 years old, ate every officer who sent the president a sandwich and a chief of staff." a delicious umbrella said.’

“The ambassador who gave the president the pickle is delicious.”

(d) Complementizer phrase

Complementizer phrases are structures that introduce subordinate clauses within sentences. They use words like 'if,' or 'because' to mark the start of these embedded clauses. Complementizer phrases enhance sentence complexity, allowing for layered meanings. They help convey relationships, conditions, or reasons in a single sentence.
We added the following rules and terminals:
ROOT → CP Comma S .
ROOT → S Comma CP .
CP → Comp S
Comp → since | if | although | because
Examples:
“If an ambassador is very elegant, every ambassador is perplexed.”
“Sally sighed, since every chief of staff is lazy.”

(e) Expand vocabulary

For more natural English we expanded our vocabulary with more words.
Verb → suggested | helped | loved
V0 → spoke
Nounp → manager
Nount → book | fruit | pizza
Prp → he | she
Vbg → kissing | taking
Nnp → John | Emily | Michael | Brian | Arthur

(f) Possessive

In this section, we support three types of possessives: possessive pronouns, possessive adjectives and possessive marker.
Possessive pronouns are pronouns that are used to indicate the ownership (possession) of something or someone by something or someone else.
Possessive adjectives are adjectives that modify a noun by identifying who has ownership or possession of it.
Possessive marker is the suffix -‘s added to a noun or pronoun to indicate ownership or possession
For <u>possessive pronouns</u> we created a new non terminal named “Posp” and we added the following rules:
VP → Vbz Posp
Posp → mine | yours
For <u>possessive adjectives</u> we added words to the Det non-terminal:
Det → ... | my | your
For <u>possessive marker</u> we added the following rule and we also added step in the post-processor that remove the whitespace before it and combined it with the proper noun.
NP →Nnp 's NN

Examples:
“Your chief of staff, Sally, is yours.”
“Although your ambassador loved with my fine floor, my proposal, 25 years old, is yours.”
“Is it true that John's sandwich is very fine?”

(g) Locations and Times

We added support for location phrase and time phrase in sentences. Times and locations come at the beginning or at the end of a sentence. Each one can come separately, or they can come together, the order doesn’t matter. If the phrase comes before the sentence its follows a comma.
The time forms we support are: weekday (Sunday, Monday...), hour (12:00...) and date (December 19...).
We added the following rules and terminals:

1. ROOT → S TLP .
2. ROOT → TLP Comma S .
3. TLP → LOCP TIMEP
4. TLP → TIMEP LOCP
5. TLP → TIMEP
6. TLP → LOCP
7. LOCP → Preplt LOC
8. LOC → Det Place
9. TIMEP → Preplt Specifictime
10. TIMEP → Prepday Dayofwee
11. TIMEP → Prepday DATE
12. DATE → Month Datenum
13. Place → office | house | park
14. Preplt → at
15. Prepday → on
16. Specifictime → Hour : Minute am | Hour : Minute pm
17. Hour → 12 | 10 | 08
18. Minute → 15 | 30 | 45 | 00
19. Dayofweek   Sunday | Thursday | Saturday
20. Datenum 19 | 4 | 12
21. Month December | February | June

Examples:
“She pickled it at my office at 10:00 am.”
“My perplexed ambassador, 18 years old, suggested that your manager sighed on Sunday.”
“At your office on June 4, your angry president, 18 years old, understood that Brian's angry pizza sighed.”
“At 12:45 pm, the apple which sighed worked with your sandwich.”