NLP Course - Assignment 2

Otmazgin, Shon
shon711@gmail.com

Rassin, Royi
isroei5700@gmail.com

December 1st, 2020

Part 0

-n flag implemented in generate.py.

Part 1

Q: 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.

A: The grammar generates long sentences due to the combination of the two rules *NP* →*NP PP* and *PP* →*Prep NP* is cyclical. Furthermore, the chances for picking the rule *NP* →*NP PP* are 50%. Thus, sequences tend to be longer and possibly infinite.

Q: The grammar allows multiple adjectives, as in: "the fine perplexed pickle". Why do the generated sentences show this so rarely? discuss.

A: In order to generate an adjective, first, we need to generate *NP* →*Det Noun* and then, *Noun* →*Adj Noun.* For a sentence to have multiple adjectives, we need it to repeat the rule *Noun* →*Adj Noun* at least twice. The grammar includes six *Noun* rules, and only one of them generates adjective. Considering that all of the rules have the same weight, the probability to generate a sequence of x adjectives is $\frac{1}{6}^x$ for instance two adjectives: $\frac{1}{6}^2 \approx 0.027$. In other words, the *Noun* →*Adj Noun* rule's weight is outweighed by the other Noun rules, so it is rarely selected.

Q: Which numbers must you modify to fix the problems in (1) and (2), making the sentences shorter and the adjectives more frequent?

A: The problem discussed in (1) can be solved by increasing the weight of *NP* →*Det Noun* to six[1]. Problem (2) can be solved by increasing the rule *Noun*

[1] there is no 'correct' number, it is personal preference

→*Adj Noun* weight to three. This way, the total score of the *Noun* rules is eight, and thus, the chance for an adjective is $\frac{3}{8}$. It is possible to increase the chances even further, but it makes the sentences appear unnatural; very long sequence of adjectives or infrequent PPs.

Q: What other numeric adjustments can you make to the grammar in order to favor a set of more natural sentences? Experiment and discuss.

A: We can make sentences appear more natural if we focus the grammar on a particular subset of words. In our case, nouns such as *'president'* and *'chief of staff'* and verbs such as *'wanted'*, *'kissed'*, and *'understood'* will be selected (specifically, increase their weight to ten). This way, there are better chances that a noun will perform a verb that makes sense. There are better chances to have sentences such as: *the president kissed the chief of staff.* In addition, determiners like *'a'* or *'the'* are more suitable than *'every'* to the verbs and nouns in the given lexicon (*'every president kissed every chief of staff'* sounds odd), their weights were increased to five. Finally, *'fine'* is a more natural-sounding adjective than *'pickled'*, *'perplexed'* and *'delicious'* and fits a broader category of nouns so its weight is ten.

> **Note**
>
> Attached files: grammar1, grammar1.gen

Part 2

Generalization

- Conjunctions (Cc): The grammar includes *'or'* conjunction.
- Prepositional Phrases (Prep): The grammar includes the prepositions *'over'* and *'of'*.
- Nouns (Noun): The grammar includes: *'rainbow'*, *'united states'*, *'man'*, and *'lady'*.
- Adjectives (Adj): The grammar includes: *'sweet'*, *'smart'*, *'hungry'*, *'dirty'*, *'artificial'*, and *'new'*.

Modifications

The most significant modifications are splitting the VPs to three categories and the NPs to two categories. The idea behind the VPs' split is to differentiate between VBGs (Vbg) such as 'eating', that require an auxiliary before them, intransitive-verbs (Verb0), that do not require an object after them, and transitive-verbs (Verb1), that do require a direct object.
This way, we can create sentences such as:

(i) *sally is eating the sandwich* (Vbg)

(ii) *the president sighed* (Verb0)

(iii) *the president kissed sally* (Verb1)

While not over-generating:

(i) *sally sighed the president* - Verb0 can't have a direct object

(ii) *The president kissed* - Verb1 needs a direct object

(iii) *sally is kissed the president* - kissed should not be preceded by 'is'

The NPs splitted to two main categories: NP1 and NP2. NP1 is focused on nouns that require a determiner, while NP2 focuses on pronouns (Prp) and proper-nouns (Nnp).
As a result, we avoid:

(i) *the sally is eating a sandwich*

(ii) *president kissed the Sally*

(iii) *a it kissed president*

> **Note**
>
> *'eating'* can also play a role of an intransitive verb because *"sally is eating"* is a valid sentence, and we indeed generate it.

> **Note**
>
> Prps (i.e. *Sally*) are lower-cased because terminals follow that convention.

Conflicting sentences

(b) *sally and the president wanted and ate a sandwich .*
(h) *sally is lazy .*
(i) *sally is eating a sandwich .*

The above sentences can interact in a way that creates ungrammatical sentences. One reason for that is adding another entity and not taking plurality into account. For instance, our rules neglect applying *'are'* instead of *'is'*.
(b) + (h) may result in: *sally and the president is lazy.*

It can be generated with the following rules:
1. S →NP Aux Adj
2. NP →NP Cc NP
3. NP →NP2
4. NP2 →Nnp
5. Nnp →sally

6. Cc →and
7. NP →NP1
8. NP1→Det Noun
9. Det →the
10. Noun →president
11. Aux →is
12. Adj →lazy

Similarly, (b) + (i) may result in: *sally and the president is eating a sandwich.*

It was also possible to end up with: *sally and the president wanted and is eating a sandwich.* However, we managed to handle this situation by differentiating between a transitive verb, and a Vbg verb, such as *'eating'*. We did so by distinguishing between Verb1, Verb0, and Vbg. Our VP can lead to three types of verbs: VP0 (intransitive verb-phrase), VP1 (transitive verb-phrase) and Vbg (preceded by an auxiliary). Consequently, we can control the type of verbs which are applied when composing long sentences with conjunctions. Specifically, the differentiating rule is:

(i) *VP VP0* - there is no going back from VP0 to VP)

(ii) *VP VP1* - there is no going back from VP1 to VP)

(iii) *VP Aux Vbg (NP)* - we have two versions of this rule, '( )' marks an option

Weights

The weights were adjusted so that the CFG will favor brief sentences. With that mind, rules like:

(i) *S →S Cc S*

(ii) *NP →NP Cc NP*

(iii) *NP1 →NP1 PP*

(iv) *VP1 →Verb1 SBAR*

(v) *VP1 →Verb1 NP SBAR*

were set with lower weights.
In contrast to rules like:

(i) *S →NP VP*

(ii) *S →NP Aux Adj*

(iii) *S →NP Aux NP*

(iv) *NP1 →Det Noun*

(v) *NP →NP1*

(vi) *NP →NP2*

(vii) *VP1 →Verb1 NP*

Which were given higher weights.

> **Note** Attached files: grammar2, grammar2.gen

Part 3

-t flag implemented in generate.py.

> **Note**
>
> Attached files: part3.gen

Part 4

We chose the phenomena (c) and (d).

Relative clauses (c)

The main focus of (c) is understanding how relative-clauses work, and that we need them to describe nouns. We can see this concept in action with a sentence like: *the pickle kissed the president that ate the sandwich.* Who did the pickle kiss? The pickle kissed the president that ate the sandwich. Relative-clauses are here to help us specify who or what we are referring to. Because of that, when we refer to someone like *'sally'* or *'it'*, a relative-clause is not necessary, we already know who the writer is referring to (theoretically it is possible that there are two Sallys, but that's not relevant to our analysis). Consequently, we defined new rules:

(i) *NP1 →NP1 RELCLAUSE*

(ii) *RELCLAUSE →that VP*

(iii) *S →NP1 RELCLAUSE*

Luckily, sentences such as: *the pickle kissed the sandwich that the president ate* were already captured by our grammar by SBAR (*SBAR →that S*) that was defined in part-2.

An unexpected complication

Despite that our grammar generated all of the sentences from previous sections (including their version as a relative-clause), it also generated some with the same structure as: *'is it true that the president that sighed?'* or *'is it true that the president that wanted sally?'* and other sentences that feel incomplete. The reason for that is because the sentence begins by posing a question, but proceeds to state a statement. To address the issue, we defined a set of rules that is capable of continuing the sentence in the same note. And specifically, to deal with the problem above, we defined the rule: *IIT-S →IIT-NP1 RELCLAUSE VP* This way, we add a verb and successfully pose a question: *is it true that the president that sighed ate a sandwich?* Because this rule is not applicable to the rest of the ROOTs, we defined a specific set (prefixed with IIT; is it true) that fits a root that begins by posing a question. Unfortunately, it also means some duplicated rules.

WH-word questions (d)

(d) was somewhat easier to define, and was mostly covered by defining a new root rule: *ROOT →i wonder WH-S .* a few WH-S rules, and *WH-VERB1* that assures us we do not pose questions like: *'i wonder what the president sighed'*, or *'i wonder what the president wanted i wonder who is lazy'*, which can happen since we have the rule: *VP1 →Verb1 SBAR* and many other ungrammatical sentences that do not fit the structure of embedded questions. We also introduced a rule that was not part of the assignment but seemed simple to include: *WH-S →where NP is* which generates sentences such as: *i wonder where sally is .*

The commonalities between (c) and (d)

'that' is a relative pronoun, and we use it to begin an adjective-clause. Specifically, we use it to describe the noun that appears before it. In (c), we introduce our grammar to relative-clauses, but are limited to statements, while in (d) we use WH-words (such as who and that) to pose (embedded) questions.

Merging

Because of the inherent difference between the two tasks: both were solved by introducing new ROOT rules. The only problem that could have arose was if the relative-clause rule and the embedded-questions rules would have interacted badly. But because we defined the WH-VERB1 rules, that problem was averted. All the while our grammar is still capable of generating embedded-questions with relative-clauses.

> **Note**
>
> Attached files: grammar4

Part 5

We decided to model two additional phenomena. The first, distinguishing between nouns that can perform a verb, Actors, and those who cannot, Objects. Our rules were adjusted to generate sentences accordingly. This way, we get sentences such as: *'the president is eating a pickle'* but not: *'the pickle is eating a pickle.* Furthermore, we are now also capable of generating: *'the president is smart'* but not: *'the pickle is smart'*, and in similar fashion, *'the pickle is pickled'* but not: *'the president is pickled'.* The adjustment entailed breaking down Nouns to actor-nouns (Act) and object-nouns (Obj), adjectives to actor-adjectives (*A-adj*) and object-adjectives (*O-adj*), and introducing *A-NP*, *A-NP1*, and *O-NP*. Maintaining the generation of previous sentences while introducing this feature and not breaking the structure we built in part-2 and part-4 proved to be a challenge and significantly increased the number of rules we have. Technically, the *NP* we defined in part-2 still proved to be useful in cases where a verb is not preceded by an *NP*. However, simply applying *NP1* as an actor would still not distinguish between actors and objects. Thus, *A-NP* was defined to create that distinction and assure *NP2s* are not followed by a *PP* or relative-clauses, while other actors still do.

The second phenomena is applying *'are'* when we have plurality. This is expressed by our ability to generate sentences such as: *'the president and the chief of staff are eating'*, and *'the president and the chief of staff are sweet'.* We defined three new rules:

(i) *S →A-NP Cc A-NP are Vbg*

(ii) *S →A-NP Cc A-NP are A-adj*

(iii) *S →O-NP Cc O-NP are O-adj*

Unfortunately, we did not have enough time to implement sentences such as *'the man and the president are a sandwich'* and stop generating *'the man and the president is eating'.*

Future Work

We are confident that with a little more time we would have completed implementing the second phenomena in part-5. That is, modifying existing rules and introducing new ones entails rigorous testing to indeed verify previous requirements are still met. Furthermore we would have significantly reduced the number of rules we have in our grammar (by finding an intelligent way to generalize the *NP* rules and be rid of the almost-duplicate rules we have among *NP*, *A-NP*, *O-NP*, *IIT-NP*, and the likes).

> **Note**
>
> Attached files: grammar5