BBM497: Introduction to Natural Language Processing Lab. **(Due: 04/05/2020)**

Assignment #3

Instructor: Necva Bölücü *Name:* MELTEM TOKGÖZ, *Netid:* 21527381

INTRODUCTION

We were given a data set containing rules, words and their type. In this project, I put the given rules and words into dictionaries. I describe the structure of my dictionaries in detail in section one. Then, using the given words, I created random sentences of the selected length and saved them in a file named output.txt. I applied the cyk parser algorithm to test whether these sentences I created are gammatically correct or false. I will explain the structure of Cyk parser algorithm and my results below.

1. **Language Generation with CFG**

   Firstly, i read the input file with reference to the comment lines, so comment lines should be given the same in other input files. After reading the file, I created 2 dictionaries. The first is for words. I used the words as key and the types as values. The second dictionary is for rules. I used the right part of the rules, (second element and after) of the line in the input file as key and the left side of the rule, as values.

```
word_dict = { 'word' : 'word_type' ...}
example : {'ate': 'Verb', 'floor' : 'Noun' ...}
_____________________________________________
rule_dict = {'right side in rule': 'left side in rule' ...}
example = {'NP VP': 'S', 'Verb NP': 'VP' ...}
```

Figure 1: CYK Algorithm

   Then, using the random.choice () I produced sentences of the desired length from the given words in the random-sentence function. I saved the sentences I produced in a file called output.txt. Some examples of these sentences are given in the picture below.

```
1  in kissed need wanted pickled
2  on with washed sandwich floor
3  like this under pickled pickled
4  every kissed fine wanted kissed
5  to floor want this that
6  from washed ate beautiful with
```

Figure 2: Random Sentece

   Note: You can try with any length and sentence as you want by changing the sent-length and sent-count parameters.

2. **Parsing Sentences with CYK Parser**

   The first thing I did when creating the Cyk parser algorithm was to create a mxm matrix as long as the length of the sentence. I used this matrix as a cyk table, and made a transaction just to be lower triangle matrix. The matrix I made in Figure 3 is shown in detail. In the Cyk algorithm, I noticed that the probabilities controlled for each matrix element consisted of pairs and these pairs were elements of the mastris. I then examined how these controlled pairs changed and were selected. And I reached a rule like figure 4. If we call the controlled element of the matrix X [i, j], the pairs that were controlled changed with a rule like in figure 4. So, i coded cyk parser algorithm according to this rule.

In the third figure, for a sentence with 5 elements, it is written which pairs are checked in which element. Black writings refer to pairs checked. Red writings refer to the element of the matrix. The pairs written here are in accordance with my rule in figure fourth.

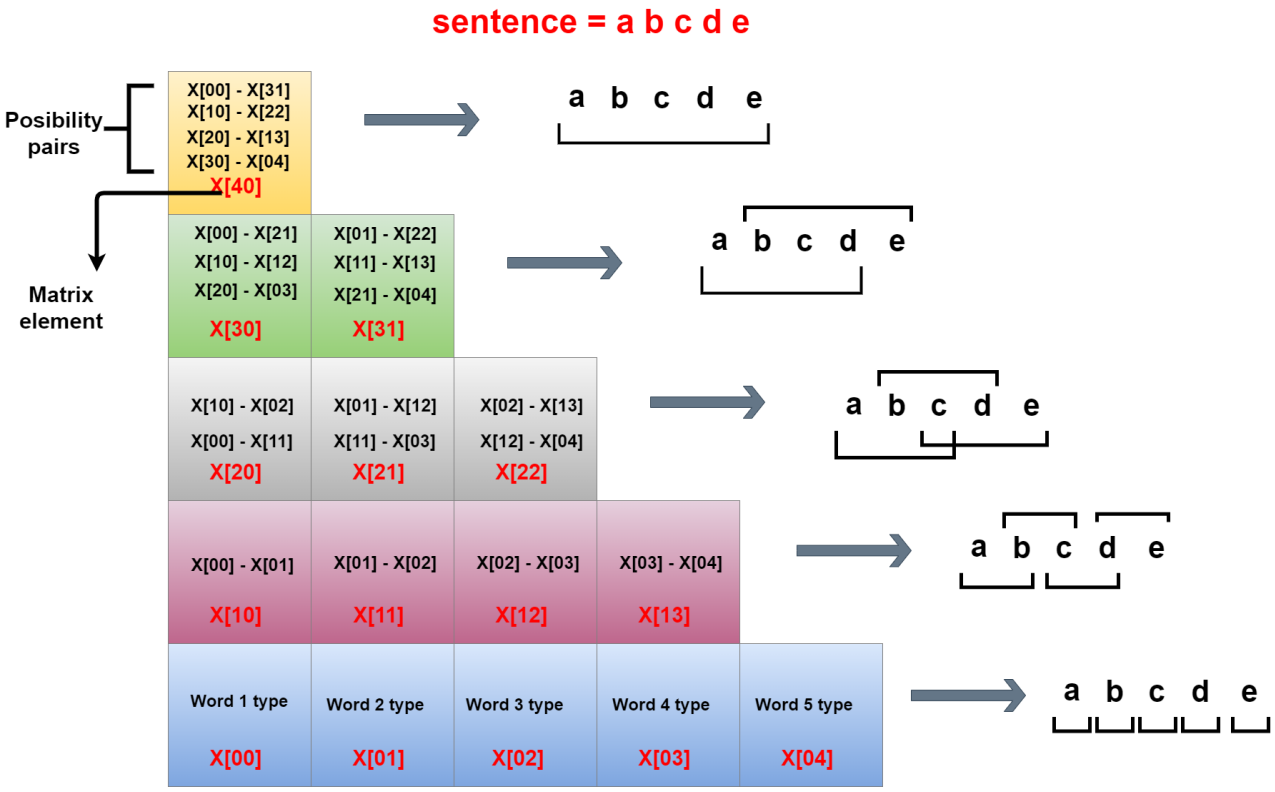

Figure 3: CYK TABLE

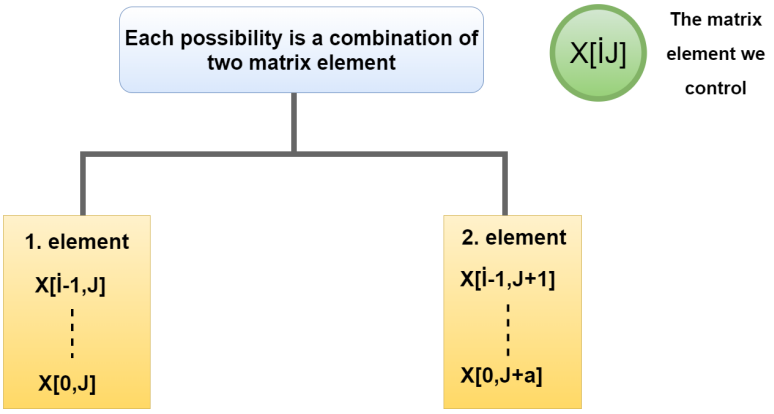

Figure 4: CYK Algorithm

Note : Since the pronoun is already equal to 'NP' according to the rules, I recorded the noun words as both noun and NP in the table, since this is also the case in the examples we did in the lesson before.

3. **Evaluation**

If the Cyk parser algorithm can reach S at the top of the table, this means a grammatically correct sentence. If can't reach S, it means a grammatically wrong sentence. In Figure 5, example sentences, results and cyk tables created by the code are given.

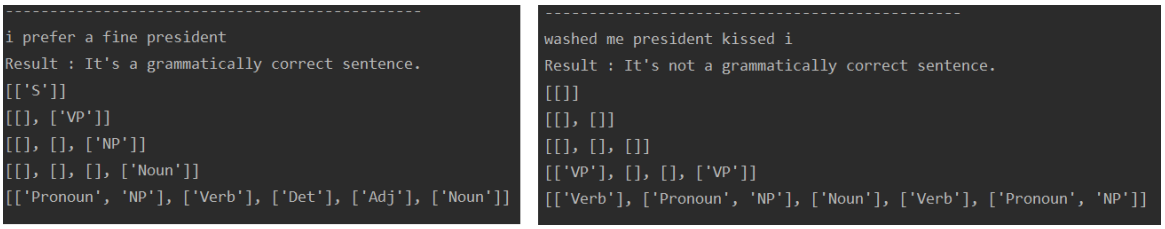

Figure 5: Example Result and Table

4. **Error Analysis**

   I have to say that when we create the sentences completely random according to the words, the probability of a grammatically correct sentence that to be quite low. If I had created sentences according to the rule set, it would be more likely to be the correct sentence, but at the very beginning I started like this and I did not change it because you stated that you would accept it like this.

5. **Analysis**

   Since there is no accuracy calculation here, I tried to observe how the cyk algorithm works in different situations. First of all, I want to give examples in some of my results.

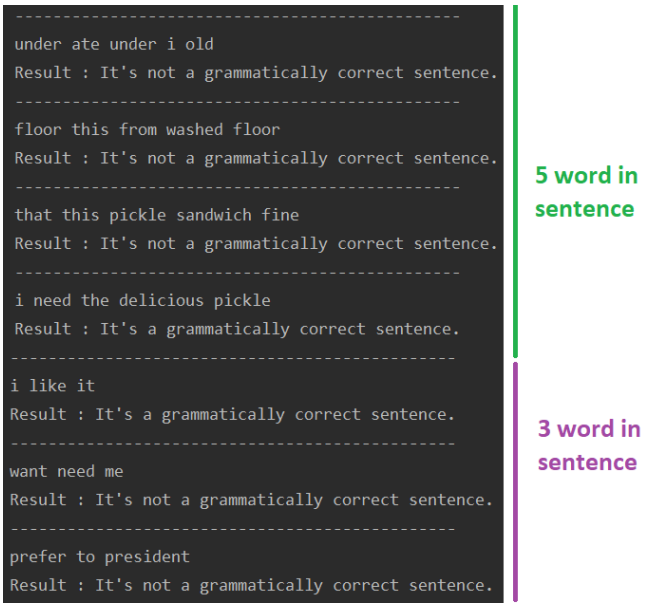

Figure 6: Some Results

- Sentence Length : I have observed in my tests that is, the short sentences are slightly more likely to be grammatically correct than the long sentences. In sentences with many words, it is unlikely to be true because more rules need to be followed.
- Since the algorithm works only according to the rule set given in the input file, every sentence that is correct in English may not be correct here.