Experiment No - 03

Aim:- To perform N-gram Language model with nltk.

Theory:-

N-gram is a sequence of the N-words in the modeling of NLP. Consider an example of the statement for modeling. “I love reading history books and watching documentaries”. In one-gram or unigram, there is a one-word sequence. As for the above statement, in one gram it can be “I”, “love”, “history”, “books”, “and”, “watching”, “documentaries”. In two-gram or the bi-gram, there is the two-word sequence i.e. “I love”, “love reading”, or “history books”. In the three-gram or the tri-gram, there are the three words sequences i.e. “I love reading”, “history books,” or “and watching documentaries” [3]. The illustration of the N-gram modeling i.e. for N=1,2,3 is given below in Figure 2 [5].

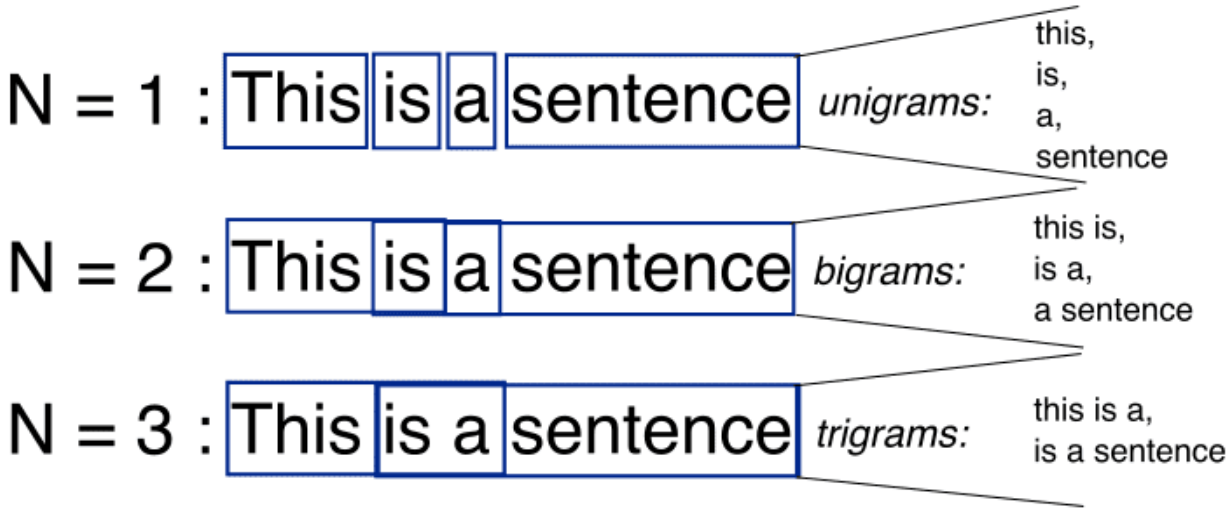


Figure 2 Uni-gram, Bi-gram, and Tri-gram Model

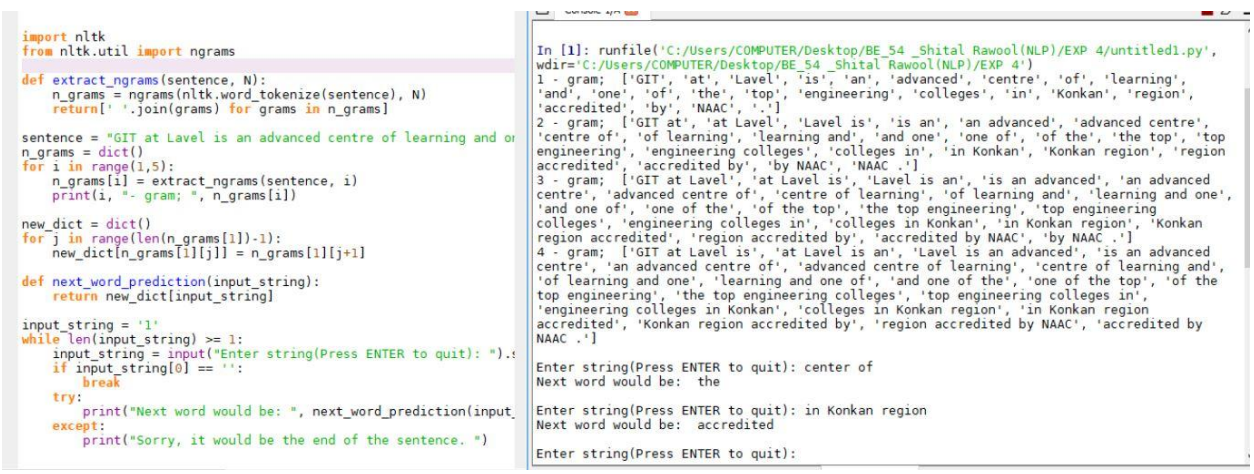
For N-1 words, the N-gram modeling predicts most occurred words that can follow the sequences. The model is the probabilistic language model which is trained on the collection of the text. This model is useful in applications i.e. speech recognition, and machine translations. A simple model has some limitations that can be improved by smoothing, interpolations, and back off. So, the N-gram language model is about finding probability distributions over the sequences of the word. Consider the sentences i.e. "There was heavy rain" and "There was heavy flood". By using experience, it can be said that the first statement is good. The N-gram language model says that the "heavy rain" occurs more frequently than the "heavy flood". So, the first statement is more likely to occur and it will be then selected by this model. In the one-gram model, the model usually relies on which word occurs often without pondering the previous words. In 2-gram, only the previous word is considered for predicting the current word. In 3-gram, two previous words are considered.

Code:-

```
import nltk
from nltk.util import ngrams

def extract_ngrams(sentence, N):
    n_grams = ngrams(nltk.word_tokenize(sentence), N)
    return[' '.join(grams) for grams in n_grams]

sentence = "GIT at Lavel is an advanced centre of learning and one of the top engineering
colleges in Konkan region accredited by NAAC. "
n_grams = dict()
for i in range(1,5):
    n_grams[i] = extract_ngrams(sentence, i)
    print(i, "- gram; ", n_grams[i])
```

```
new_dict = dict()
for j in range(len(n_grams[1])-1):
    new_dict[n_grams[1][j]] = n_grams[1][j+1]

def next_word_prediction(input_string):
    return new_dict[input_string]

input_string = '1'
while len(input_string) >= 1:
    input_string = input("Enter string(Press ENTER to quit): ").split(' ')
    if input_string[0] == '':
        break
    try:
        print("Next word would be: ", next_word_prediction(input_string[-1].strip()))
    except:
        print("Sorry, it would be the end of the sentence. ")
```

Output:-



Conclusion:

Thus We have successfully implemented N -gram model