Experiment No.5
Implement Bi-Gram model for the given Text input
Date of Performance:
Date of Submission:

Aim: Implement Bi-Gram model for the given Text input Objective: To study and implement N-gram Language Model.

Theory:

A language model supports predicting the completion of a sentence.
Eg:

- Please turn off your cell _____
- Your program does not ______

Predictive text input systems can guess what you are typing and give choices on how to complete it.

N-gram Models:
Estimate probability of each word given prior context. P(phone | Please turn off your cell)

- Number of parameters required grows exponentially with the number of words of prior context.
- An N-gram model uses only N1 words of prior context.
  - ○ Unigram: P(phone)
  - ○ Bigram: P(phone | cell)
  - ○ Trigram: P(phone | your cell)

- The Markov assumption is the presumption that the future behavior of a dynamical system only depends on its recent history. In particular, in a kth-order Markov model, the next state only depends on the k most recent states, therefore an N-gram model is a (N1)-order Markov model.

N-grams: a contiguous sequence of n tokens from a given piece of text

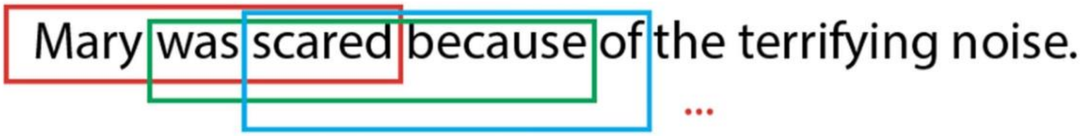

Fig. Example of Trigrams in a sentence

Parts of Speech

Tag|Meaning|English Examples

ADJ|adjective|new, good, high, special, big, local

ADP|adposition|on, of, at, with, by, into, under

ADV|adverb|really, already, still, early, now

CONJ|conjunction|and, or, but, if, while, although

DET|determiner, article|the, a, some, most, every, no, which

NOUN|noun|year, home, costs, time, Africa

NUM|numeral|twenty-four, fourth, 1991, 14:24

PRT|particle|at, on, out, over per, that, up, with

PRON|pronoun|he, their, her, its, my, I, us VERB|verb|is,

say, told, given, playing, would .|punctuation marks|. , ; !

X|other|ersatz, esprit, dunno, gr8, univeristy

```
text = "TON 618 (short for Tonantzintla 618) is a hyperluminous, broad-absorption-line, radio-loud quasar and Lyman-alpha blob located
```

ne Importing necessary dependencies

```
import nltk
from nltk.tokenize import word_tokenize
```

Word Tokenization

```
nltk.download('punkt')
words =
 word_tokenize(text)
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

Parts of Speech Tagging

```
nltk.download('universal_tagset')
nltk.download('averaged_perceptron_tagger')
tagged_words = nltk.pos_tag(words, tagset =
'universal')
```

```
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
```

```
tagged_words
```

```
[('TON', '.'),
 ('618', 'NUM'),
 ('(', '.'),
 ('short', 'ADJ'),
 ('for', 'ADP'),
 ('Tonantzintla', 'NOUN'),
 ('618', 'NUM'),
 (')', '.'),
 ('is', 'VERB'),
 ('a', 'DET'),
 ('hyperluminous', 'ADJ'),
 (',', '.'),
 ('broad-absorption-line', 'ADJ'),
 (',', '.'),
 ('radio-loud', 'ADJ'),
 ('quasar', 'NOUN'),
 ('and', 'CONJ'),
 ('Lyman-alpha', 'NOUN'),
 ('blob', 'NOUN'),
 ('located', 'VERB'),
 ('near', 'ADP'),
 ('the', 'DET'),
 ('border', 'NOUN'),
 ('of', 'ADP'),
 ('the', 'DET'),
 ('constellations', 'NOUN'),
 ('Canes', 'NOUN'),
 ('Venatici', 'NOUN'),
 ('and', 'CONJ'),
 ('Coma', 'NOUN'),
 ('Berenices', 'NOUN'),
 (',', '.'),
 ('with', 'ADP'),
 ('the', 'DET'),
 ('projected', 'VERB'),
 ('comoving', 'NOUN'),
 ('distance', 'NOUN'),
 ('of', 'ADP'),
 ('approximately', 'ADV'),
 ('18.2', 'NUM'),
 ('billion', 'NUM'),
 ('light-years', 'NOUN'),
 ('from', 'ADP'),
 ('Earth', 'NOUN'),
 ('.', '.')]
```

```
for t in tagged_words:
    print(t)
```

```
('TON', '.')
('618', 'NUM')
('(', '.')
('short', 'ADJ')
('for', 'ADP')
('Tonantzintla', 'NOUN')
('618', 'NUM')
(')', '.')
('is', 'VERB')
('a', 'DET')
('hyperluminous', 'ADJ')
(',', '.')
('broad-absorption-line', 'ADJ')
(',', '.')
('radio-loud', 'ADJ')
('quasar', 'NOUN')
('and', 'CONJ')
('Lyman-alpha', 'NOUN')
('blob', 'NOUN')
('located', 'VERB')
('near', 'ADP')
```

```
('the', 'DET')
('border', 'NOUN')
('of', 'ADP')
('the', 'DET')
('constellations', 'NOUN')
('Canes', 'NOUN')
('Venatici', 'NOUN')
('and', 'CONJ')
('Coma', 'NOUN')
('Berenices', 'NOUN')
(',', '.')
('with', 'ADP')
('the', 'DET')
('projected', 'VERB')
('comoving', 'NOUN')
('distance', 'NOUN')
('of', 'ADP')
('approximately', 'ADV')
('18.2', 'NUM')
('billion', 'NUM')
('light-years', 'NOUN')
('from', 'ADP')
('Earth', 'NOUN')
('.', '.')
```

Conclusion:

A bi-gram model is a language model that examines sequences of two adjacent words in a given text. By analyzing word pairs, it captures some level of contextual information. However, it has limitations, such as ignoring longer-range dependencies and lacking semantic understanding. Bigram models can be useful for simple tasks like text prediction or basic sentiment analysis, but for more advanced NLP applications