ST.FRANCIS INSTITUTE OF TECHNOLOGY
Mount Poinsur, S.V.P. Road, Borivli (West), Mumbai - 400103
Computer Engineering Department

Academic Year: 2021-2022 **Class/Branch**: BE COMP
Subject: DLO 8012 Natural Language Processing **Semester**: VIII

Name: Tanmay Desai Roll No:17 PID: 182025 BE CMPN A

Experiment No 04

N gram Modelling

Aim : To implement the Ngram model from a text corpus and do adjacent word prediction in Python

Theory :

The idea of word prediction is to predict the next word from the previous N−1 words using probabilistic models called N−gram models. The estimators like N−grams that assign a conditional probability to possible next words can be used to assign a joint probability to an entire sentence. Whether estimating probabilities of next words or of whole sequences, the N−gram model is one of the most important tools in speech and language processing.

N-gram models require that we do the kind of tokenization or text normalization: separating out punctuation, dealing with abbreviations like m.p.h., normalizing spelling, and so on. Our goal is to compute the probability of a word w given some history h, or P(w|h). Suppose the history h is “its water is so transparent that” and we want to know the probability that the next word is the: P(the|its water is so transparent that).

One way to compute probability is to estimate it from relative frequency counts. For example, we could take a very large corpus, count the number of times we see the water is so transparent that, and count the number of times this is followed by the. This may be given as follows:

$$P(\text{the}|\text{its water is so transparent that}) = \frac{C(\text{its water is so transparent that the})}{C(\text{its water is so transparent that})}$$

The intuition of the N-gram model is that instead of computing the probability of a word given its entire history, approximate the history by just the last few words. The bigram model, for example, approximates the probability of a word given all the previous words $P(w_n|w_1^{n-1})$ by using only the conditional probability of the preceding word $P(w_n|w_{n-1})$. We can generalize the bigram (which looks one word into the past) to the trigram (which looks two words into the past) and thus to the N-gram (which looks N-1 words into the past).

Programming Exercises

1. Write a Python program to do word prediction using N-gram Model (Use bigrams and trigrams) given a corpus in .txt format.

Colab File:
https://colab.research.google.com/drive/1omKttYfMUOnRwHlRNvTS1_FRT3olJ-8K?usp=sharing

Academic Year: 2021-2022 **Class/Branch**: BE COMP
Subject: DLO 8012 Natural Language Processing **Semester**: VIII

Mini Project Exercises

1. Do requirement analysis and design of miniproject and make a report based on Major Project Guidelines given.