Experiment No. 1(A)

Name :- Indraneel Adak

Roll No. :- 16

Class :- B.E.

Aim :- To execute a program for tokenization of a sentence or paragraph.

Theory :- Tokenization is the process of tokenization or spliting a string, text into a list of token. one can think of token as parts like a word is token in a sentence & a sentence is a token in a paragraph.

Here, tokens can be either words, characters or subwords. Hence tokenization can be broadly classified into 3-types - word, character & subword tokenization. The most common way of forming token is based on space i.e. assuming as delimiter, the tokenization of sentence results in 3 tokens - never give up. As each token is a word. It becomes an example of word tokenization.

Similarly tokens can be either characters or subwords e.g let us consider "smarter"

character token s-m-a-r-t-e-r

subword tokens smart-er

How to execute tokenization using ~~NT~~ NLTK?

For this we first download NLTK in our python environment and then import the sent-tokenize, word-tokenize from the NLTK.

tokenize.class sent-tokenize can be used for work tokenization of given or paused sample test.

Conclusion :- Thus we have successfully executed tokenization using NLTK library in python.

Experiment No. 1 (B)

Name :- Indraneel Adak

Roll No. :- 16

Class :- B.E.

Aim :- TO execute a program from removing stopword from sentence.

Theory :- A stopword is a commonly used word (such as "the", "a", "an", "in") that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

Stopword are the english words which do not add much meaning to a sentence stopword are often removed from the text before training deep learning & ML models since stopword occur in abundance, hence providing little to no unique information that can be used for classification or clustering.

Removing stopword using NLTK library.

The NLTK library is one of the oldest and most commonly used python libraries for NLP. NLTK supports stopword removal and you can find the list of the stopword in the corpus module.

TO remove stop words from a sentence you can divide your ~~tes~~ text into words & then remove the word if it exists in the list of stopwords provided by the NLTK.

We first import the stopwords collection from the NLTK corpus module. We ~~co~~ then

Create a variable text, which contains a simple sentence.

The sentence in the text variable is tokenized. using the word_tokenize() method. Next we iterate through all the words in the text_tokens list and checks if the word exists in stopword collection or not. The tokens without stopword list is then printed.

Conclusion :- Thus we have successfully executed a program for stopword removal using NLTK library in python.

Experiment No. 2

Name :- Indraneel Adak

Roll No. :- 16

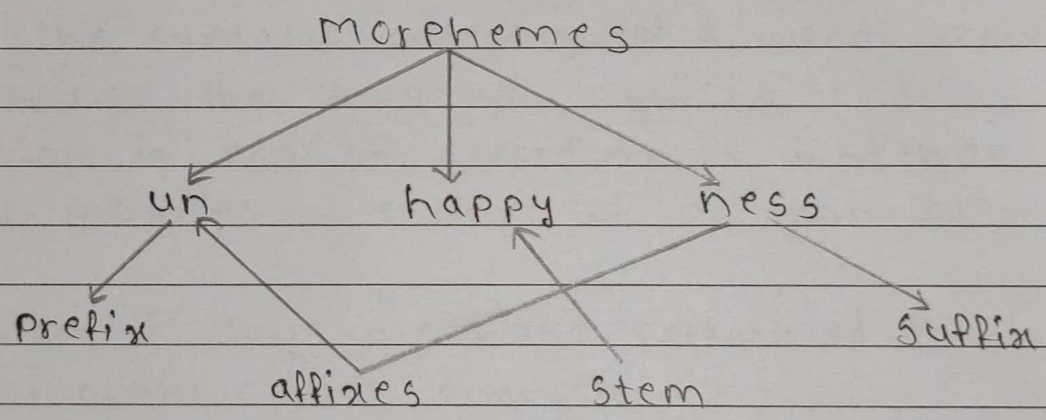
Class :- B.E.

Aim :- To demonstrate Morphological Analysis.

Theory :-

Morphology is the study of structure & formation of words. It's ~~not~~ most important unit is morpheme which is defined as the "minimal unit" of meaning.

- Consider a word like : "unhappiness". This has three parts



- Each morpheme carries certain amount of meaning.
- Un means not, ness means being in state or condition.
- Happy is free morpheme because it appears on its own as word.

*] Inflection :-

Inflection is the process of changing the form of a word so that it express information such as number, person, case, gender, tense, mood

and aspect but syntactic category of words remains unchanged. As an example, the plural form of the noun in english is usually formed from singular form by adding an s.

- car / cars
- table / tables
- dog / dogs

*) Derivation :-

As was seen above inflection doesn't change the syntatic category of a word. Derivation does change the category. Linguists classify derivation in english according to whether or not it includes a change of pronounciation.

Conclusion :- Thus we have performed the morphological analysis.

Experiment No. 3

Name :- Indraneel Adak

Roll No :- 16

Class :- B.E.

Aim :- To implement N-gram model.

Theory :- N-gram

- The general idea is that you can look each pair or triple etc of words that occur next to each other.
- On a sufficiently large corpus, your's likely is see "apple" & "red" several times but less likely to "apple red" & "~~th~~ red the".
- The co-occuring words are known as n-grams.
- n-~~gram~~ number defines how long string of words.

  Unigram → single word
  Bigram → two words
  Trigram → three words
  4-gram → four words
- NLTK has n-grams functions that refers a generator of n-grams given a tokenized sentence.
- An n-gram tagger is generalization of unigram tagger whose context is correct word together with part of speech tags of n-1 preceding tokens.
- The N-gram tagger class uses tagged training corpus to determine which part-of-speech tag is most likely for each context.

Conclusion :- Hence we studied & implemented the N-gram model.

Experiment No. 4

Name :- Indraneel Adak

Roll No :- 16

Class :- B.E.

Aim :- To implement POS tagging.

Theory :-
- Tagging is kind of classification that may be defined as the automatic assignment of description to the token. Here the descriptor called tag which represent one of the POS semantic info. & so on.
- POS tagging may defined of process of assigning one of the part-of-speech to the given word.

*] Role based POS Tagging :-
- It's one of the oldest tagging techniques.
- It uses dictionary or lexicon for getting possible tags for tagging each word.
- If word has more than one tag then it use handwritten rule to identify correct tag.
- First stage : It uses dictionary to assign each word a list of potential POS.
- Second stage : It uses large list of handwritten disambiguation to sort down the list to single POS

*] Properties :-
- These taggers are knowledge-driven tagger.
- Roles in Rule based POS built manually.
- Info coded in form of roles
- Limited numbers of roles approx 1000.
- Smoothing & language modeline defined explicitely.

*] Stochastic POS Tagging :-
- The model that includes probability can be called stochastic.
- Any number of diff. approaches to POS tagging problem can be referred to stochastic tagger.

*] Word preacency approach :
- Stochastic tagger disambiguate word based on the probability that word occurs with particular tag.
- The issue with this approach is it may yield inadmissible sequence of tags.

*] Tag sequence probabilities :
- This approach tagger calculates probability of given sequence of tag occuring.
- It is also called as n-gram approach.

*] Properties :-
- It is based on probability of tag occuring.
- It requires training corpus.
- It is the simplest POS tagging because it chooses most frequent tags associated with word in the training corpus.

Conclusion :- Thus we have successfully implemented POS tagging.

Experiment No. 5

Name :- Indraneel Adak

Roll No :- 16

Class :- B.E.

Aim :- To implement chunking.

Theory :- Chunking

- Text chunking is also reffered as shallow parsing.
- It is a task that follows POS tagging and adds more structure to sentence.
- The result is grouping of words in chunks.
- Chunk extraction is process of meaningful extraction of short phrases from the sentence.
- Chunks are made up of words, and kind of words are defined using POS tags.
- Chunking activity involves breaking down difficult text into more managable ~~pie~~ pieces.
- One of the main goal of chunking is to group words into what are known as "nounphrases".
- The idea is to group noun with the words that are in relation to them.
- In order to chunk, we combine the part-of-speech tags with regular expression.
- mainly from regular expressions, we are going to utilize the following.
  - + = match 1 or more
  - ? = match 0 or 1 repetitions.

* = match 0 or more repetitions

. = any character except a new line

Conclusion :- Hence we studied and ~~impleme~~ implemented chunking in NLP.

Experiment No. 6

Name :- Indraneel Adak

Roll No :- 16

Class :- B.E.

Aim :- To demonstrate Named Entity Recognition.

Theory :- Named Entity Recognition. (NER) :

- In any text document, there are particular term that represent specific entities that are more informative and have unique context.
- These entities are known as named entities which moves specificially refers to term that represent real-world object like people, place, organization and so on which are often denoted by proper names.
- A naive approach could be to find these by looking at noun phrases in text ~~doc~~ document.
- Named Entity Recognition (NER), also known as entity chunking / extraction is a popular technique used in the information extraction to identify and segment the named entities and classify / categorize them under various ~~pred~~ predefined classes.
- This can be a bit of challenge as idea behind NER is to have immediate ability to pull out entities like people, places, things, locations, monitory figures & more.

- There are two main options with NLTK's Named Entity Recognition:
→ Either recognize all named Entities.
→ OR, Recognize named entities as their respective types like people, places, location, etc.

Conclusion :- Hence we studied and implemented the named Entity Recognition (NER).