SATHYABAMA INSTITUTE OF SCIENCE & TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCSA 2604 NATURAL LANGUAGE PROCESSING LAB

LAB 7: CASE STUDY

AIM: The aim of this case study is to demonstrate the extraction of noun phrases from a given text using chunking, a technique in Natural Language Processing (NLP). We will utilize Python's NLTK library to implement chunking and extract meaningful noun phrases from the text.

Problem Statement:

Given a sample text, our goal is to identify and extract noun phrases, which are sequences of words containing a noun and optionally other words like adjectives or determiners. The problem involves implementing a program that tokenizes the text, performs part-of-speech tagging, applies chunking to identify noun phrases, and finally outputs the extracted noun phrases.

Objectives :

1. Tokenize the input text into words.
2. Perform part-of-speech tagging to assign grammatical tags to each word.
3. Define a chunk grammar to identify noun phrases.
4. Apply chunking to extract noun phrases from the text.
5. Display the extracted noun phrases.

Dataset:

For this case study, we will use a sample text: "The quick brown fox jumps over the lazy dog."

Approach:

The approach involves several steps to extract noun phrases from the given text using chunking in Natural Language Processing (NLP). Firstly, the input text is tokenized into individual words to prepare it for further processing. Following tokenization, each word is tagged with its part-of-speech using NLTK's pos_tag function, which assigns grammatical tags to each word based on its context. Next, a chunk grammar is defined to specify the patterns that identify noun phrases. This grammar is then utilized to apply chunking, which groups consecutive words that match the defined patterns into noun phrases. Finally, the extracted noun phrases are outputted, providing meaningful insights into the structure and content of the text. This approach allows for the identification and extraction of important linguistic units, facilitating various NLP tasks such as information extraction, text summarization, and sentiment analysis.

Program :

```
import nltk
import os

# Set NLTK data path
nltk.data.path.append("/usr/local/share/nltk_data")

# Download the 'punkt' tokenizer model
nltk.download('punkt')

# Download the 'averaged_perceptron_tagger' model
nltk.download('averaged_perceptron_tagger')

# Sample text
text = "The quick brown fox jumps over the lazy dog."

# Tokenize the text into words
words = nltk.word_tokenize(text)

# Perform part-of-speech tagging
pos_tags = nltk.pos_tag(words)

# Define chunk grammar
chunk_grammar = r"""
    NP: {<DT>?<JJ>*<NN>}  # Chunk sequences of DT, JJ, NN
"""

# Create chunk parser
chunk_parser = nltk.RegexpParser(chunk_grammar)

# Apply chunking
chunked_text = chunk_parser.parse(pos_tags)

# Extract noun phrases
noun_phrases = []
for subtree in chunked_text.subtrees(filter=lambda t: t.label() ==
'NP'):
    noun_phrases.append(' '.join(word for word, tag in
subtree.leaves()))

# Output
print("Original Text:", text)
print("Noun Phrases:")
for phrase in noun_phrases:
    print("-", phrase)
```

Output:

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
Original Text: The quick brown fox jumps over the lazy dog.
Noun Phrases:
- The quick brown
- fox
- the lazy dog
```

Result:

Chunking is a valuable technique in NLP for identifying and extracting meaningful phrases from text. In this case study, we successfully implemented chunking using Python's NLTK library to extract noun phrases from a given text. By identifying and extracting noun phrases, we gained insights into the structure and semantics of the text, which can be beneficial for various NLP applications such as information extraction, sentiment analysis, and text summarization.