SATHYABAMA INSTITUTE OF SCIENCE & TECHNOLOGY

SCHOOL OF COMPUTING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SCSA 2604 NATURAL LANGUAGE PROCESSING LAB

LAB – 1 : WORD ANALYSIS

AIM: To perform basic word analysis

PROCEDURE :

For word analysis in Python for NLP, we can use various libraries and techniques. Here's a simple example using spaCy to perform basic word analysis tasks. spaCy provides more sophisticated linguistic annotations and functionalities.

Tokenization: The process of splitting the text into individual words or tokens.

Lemmatization: Finding the base or dictionary form of a word (e.g., 'is' becomes 'be').

Dependency Parsing: Analyzing the syntactic structure of the sentence, showing relationships between words (e.g., subject, object) through dependency relations.

This code demonstrates the basic usage of spaCy to perform tokenization, lemmatization, and dependency parsing on a given text, providing insights into the linguistic structure and relationships within the text.

ALGORITHM:

1. Import spaCy: Import the spaCy library, which is an open-source natural language processing library.

2. Load Language Model: Load the English language model (en_core_web_sm) provided by spaCy, which includes a tokenizer, tagger, parser, named entity recognizer (NER), and word vectors.

3. Define Sample Text: Set a sample text that will be used for analysis. In this case, it's "Natural Language Processing is a fascinating field of study."

4. Process Text with spaCy: Pass the sample text through the loaded spaCy pipeline using nlp(text). This processes the text and generates a doc object that contains linguistic annotations and information about the text.
5. Tokenization and Lemmatization:
   Extract tokens: Get a list of all the tokens (individual words) in the text using a list comprehension [token.text for token in doc].
   Lemmatization: Get a list of lemmas (base forms of words) using [token.lemma_ for token in doc].
6. Print Tokens and Lemmas: Print the extracted tokens and their corresponding lemmas.
7. Dependency Parsing: Iterate through each token in the processed doc.Print information about each token's text, dependency relation (token.dep_), its head text (token.head.text), its head's part-of-speech (token.head.pos_), and its children ([child for child in token.children]).

PROGRAM

```
import spacy


# Load English tokenizer, tagger, parser, NER, and word vectors

nlp = spacy.load("en_core_web_sm")


# Sample text for analysis

text = "Natural Language Processing is a fascinating field of study."


# Process the text with spaCy

doc = nlp(text)


# Extracting tokens and lemmatization

tokens = [token.text for token in doc]

lemmas = [token.lemma_ for token in doc]


print("Tokens:", tokens)

print("Lemmas:", lemmas)


# Dependency parsing

print("\nDependency Parsing:")
```

```
for token in doc:
    print(token.text, token.dep_, token.head.text, token.head.pos_,
        [child for child in token.children])
```

OUTPUT:

```
Tokens: ['Natural', 'Language', 'Processing', 'is', 'a', 'fascinating', 'field', 'of', 'study', '.']
Lemmas: ['Natural', 'Language', 'Processing', 'be', 'a', 'fascinating', 'field', 'of', 'study', '.']

Dependency Parsing:
Natural compound Language PROPN []
Language compound Processing PROPN [Natural]
Processing nsubj is AUX [Language]
is ROOT is AUX [Processing, field, .]
a det field NOUN []
fascinating amod field NOUN []
field attr is AUX [a, fascinating, of]
of prep field NOUN [study]
study pobj of ADP []
. punct is AUX []
```