Practical No. 5

Write a Python program to find the sets of synonyms and antonyms of a given word using WordNet.

Class: Final Year (ECO) Batch: ________________

Roll No. ____________________ Date: _________________

Objectives

- To understand the concept of lexical databases in NLP.
- To learn how to use the WordNet corpus in NLTK.
- To extract sets of synonyms and antonyms for a given word.
- To demonstrate how semantic relationships can be utilized in NLP tasks.

Theory

WordNet is a large lexical database of English developed by Princeton University. It groups words into sets of synonyms called synsets and records relationships among them such as hypernyms, hyponyms, and antonyms.
In NLP, WordNet is widely used to expand meaning, find word similarities, and build semantic understanding in applications such as search engines and chatbots.

Stepwise Procedure

1. Import the NLTK library and download the WordNet corpus.
2. Accept a word as input from the user.
3. Retrieve all synsets of the input word using wordnet.synsets().
4. Extract synonyms by iterating through each synset and collecting lemma names.
5. Extract antonyms using the antonyms() method of lemmas.
6. Display all unique synonyms and antonyms of the word.

Conclusion

____

____

Post Lab Questions

Solve any 5

1. Define the term ‘synset’ in WordNet.
2. Explain the relationship between synonyms and antonyms in WordNet.
3. Mention one application where WordNet-based synonym extraction is used.
4. How does WordNet help in semantic similarity measurement?
5. Differentiate between hypernym and hyponym relationships.

Practical No. 6

Write a Python program to perform sentiment analysis.

Class: Final Year (ECO) Batch: ________________

Roll No. ______________________ Date: __________________

Objectives

1. To understand the concept of sentiment analysis in NLP.
2. To learn how to use TextBlob or NLTK for analyzing sentiment polarity and subjectivity.
3. To classify text as positive, negative, or neutral based on sentiment scores.
4. To apply sentiment analysis to sample datasets or user-input text.

Theory

Sentiment analysis, or opinion mining, is the process of identifying and categorizing opinions expressed in text to determine whether the attitude is positive, negative, or neutral.

It uses natural language processing techniques and pre-trained models that analyze words and phrases to compute sentiment polarity (range from -1 to +1) and subjectivity (range from 0 to 1).

Procedure

1. Import TextBlob library or NLTK's VADER sentiment analyzer.

2. Accept an input sentence or paragraph from the user.
3. Create a TextBlob object or use SentimentIntensityAnalyzer().
4. Compute the polarity and subjectivity (or compound score).
5. Interpret and print whether the sentiment is positive, negative, or neutral.
6. Test with multiple examples to observe variations in sentiment.

Conclusion

____

____

____

____

Post Lab Questions

1. What is sentiment polarity?
2. What are two libraries commonly used for sentiment analysis in Python?
3. Differentiate between polarity and subjectivity.
4. Mention one real-world application of sentiment analysis.
5. How can sentiment analysis support business intelligence?

Practical No. 7

Write a Python program to perform spelling correction in the given input text. Apply minimum edit distance between two strings for spelling correction.

Class: Final Year (ECO) Batch: ________________

Roll No. ____________________ Date: _________________

Objectives

1. To understand the concept of spelling correction in NLP.
2. To learn how to compute the minimum edit distance between two strings.
3. To implement automatic spelling correction using dynamic programming.
4. To explore applications of edit distance in information retrieval and text correction.

Theory

Minimum Edit Distance (MED) is the minimum number of operations required to transform one string into another. Operations include insertion, deletion, and substitution.

In spelling correction, the MED algorithm is used to find the closest valid word to a misspelled one based on minimum changes required.

Stepwise Procedure

Initialize two strings

Let the source word be **S1** (e.g., "kitten")

Let the target word be **S2** (e.g., "sitting")

Create a matrix

Construct a matrix D with dimensions (len(S1)+1) × (len(S2)+1)

Rows represent characters of S1, and columns represent characters of S2.

Initialize base cases

Set the first row values as D[0][j] = j for all columns j (cost of inserting j characters).

Set the first column values as D[i][0] = i for all rows i (cost of deleting i characters).

Fill the matrix

For each cell (i, j):

If the characters **S1[i-1] == S2[j-1]**,then cost = 0

Else,
cost = 1 (substitution required)

Compute the minimum cost using:

```
D[i][j] = min(
    D[i-1][j] + 1,          # Deletion
    D[i][j-1] + 1,          # Insertion
    D[i-1][j-1] + cost      # Substitution
)
```

Continue filling

Repeat the above step for all positions in the matrix until it's completely filled.

Trace the result

The final cell D[len(S1)][len(S2)] gives the **Minimum Edit Distance** between the two words.

(Optional) Backtracking

To identify which operations were performed (insert, delete, substitute), backtrack from the last cell to the first cell following the minimum path.

Conclusion

Post Lab Questions

1. Define minimum edit distance with an example.
2. List the three basic edit operations considered in MED.
3. Explain how MED can be used for spelling correction.
4. What is dynamic programming, and how is it used in MED calculation?
5. Name one NLP library that provides a built-in function for edit distance.

Practical No. 8

Write a Python program to perform conversion of given input text to speech using library function.

Class: Final Year (ECO)	Batch: ________________
Roll No. ____________________	Date: _________________

Objectives

1. To understand the concept of text-to-speech (TTS) conversion.
2. To explore Python libraries that support speech synthesis such as gTTS and pyttsx3.
3. To implement a Python program that converts input text into audio speech.
4. To study applications of speech synthesis in assistive technology and chatbots.

Theory

Text-to-Speech (TTS) is a process that converts digital text into spoken voice output. It uses speech synthesis techniques to generate audio output that mimics human speech.

Python provides libraries like gTTS (Google Text-to-Speech) and pyttsx3 which can be used offline or online to perform this task easily.

Stepwise Procedure

1. Install the required TTS library (gTTS or pyttsx3).
2. Accept input text from the user.
3. Initialize the TTS engine and set properties such as rate and voice.
4. Convert the input text to speech and save it as an audio file.
5. Play the generated audio file to verify output.

Conclusion

____

____

____

____

Post Lab Questions

1. Differentiate between gTTS and pyttsx3 libraries.
2. What is speech synthesis?
3. List two practical applications of text-to-speech conversion.
4. How does gTTS convert text into voice?
5. What parameters can be adjusted in a TTS engine?