Indian Institute of Information Technology Surat

Lab Report on
Natural Language Processing (CS 601) Practical

Submitted by

[RAHUL KUMAR SINGH] (UI21CS44)

Course Faculty

Mrs. Nidhi Desai

Department of Computer Science and Engineering
Indian Institute of Information Technology Surat
Gujarat-394190, India

Jan-2024

Lab No: 4

Aim:

To demonstrate the use of the Treebank and WordNet library in NLTK by performing sentence tagging and tokenization, and exploring the functions available in WordNet.

Description:

Treebank: A corpus of text annotated with syntactic structure, used for training and evaluating parsing models.

WordNet: A lexical database that groups words into synsets, providing semantic relations and definitions for computational linguistics.

POS Tagging: The sentence is tokenized and tagged using NLTK's *`word_tokenize`* and *`pos_tag`* functions.

Functions Available in WordNet:

- **synsets(word)**: Retrieves all synsets (senses) for a given word.
- **definition()**: Provides the definition of a specific synset.
- **lemmas()**: Returns the list of lemmas (base forms of a word) for a synset.
- **hypernyms()**: Shows general concepts related to the word.
- **hyponyms()**: Displays more specific categories related to the word.
- **part_meronyms()**: Retrieves parts of a concept.
- **member_holonyms()**: Finds collections to which a concept belongs.

POS Tags:

- **Sentence**: The sentence as a tokenized string.
- **Noun**: Lists all nouns in the sentence (e.g., "cat", "mat").
- **Verb**: Lists all verbs in the sentence (e.g., "sat").
- **Adjective**: Lists all adjectives (e.g., "beautiful").
- **Adverb**: Lists all adverbs (e.g., "quickly").
- **Determiner**: Lists all determiners (e.g., "the").
- **Preposition**: Lists all prepositions (e.g., "on").
- **Pronoun**: Lists all pronouns (e.g., "he").
- **Conjunction**: Lists all conjunctions (e.g., "and"").
- **Modal**: Lists modal auxiliary verbs (e.g., "will").

Output:

- **Treebank POS Tagging**: Lists the tokens along with their tags using the Treebank tagger.
- **WordNet-based Tagging**: Lists the tokens along with their WordNet-based POS tags.
- **Time Taken**: Displays the time taken by each approach.
- **Precision**: Compares the POS tagging accuracy between the two methods.

Source Code:

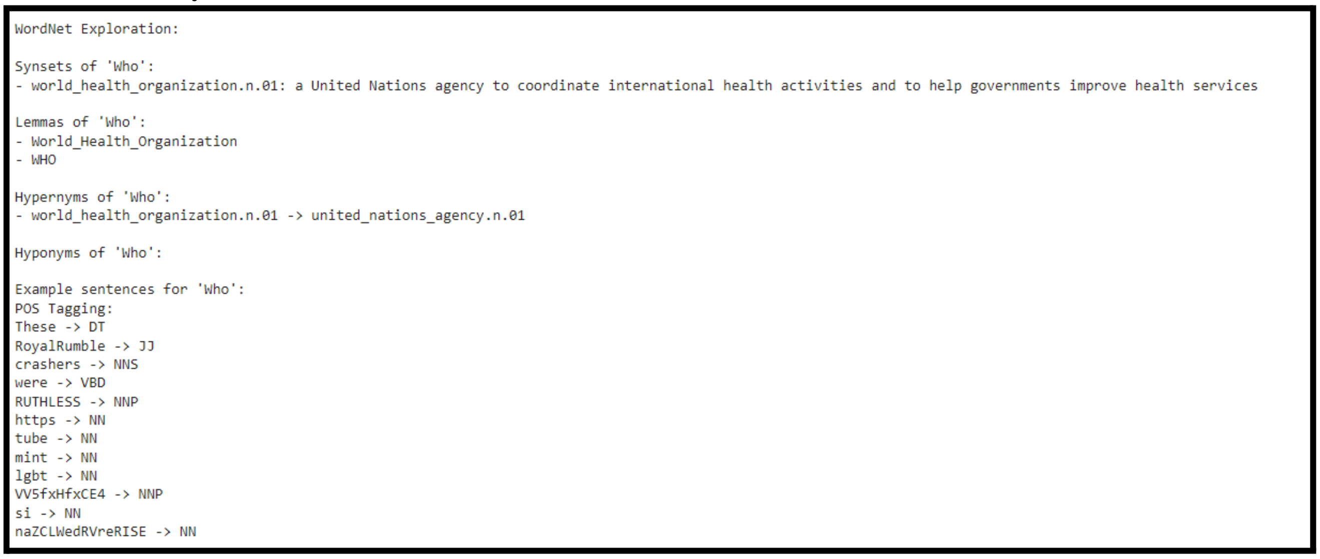
```
import nltk
from nltk.corpus import wordnet as wn
from nltk.tokenize import word_tokenize
from nltk import pos_tag
from time import time

nltk.download('treebank')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('omw-1.4')

pos_full_form = {
    'CC': 'Coordinating conjunction', 'CD': 'Cardinal digit', 'DT':
'Determiner',
    'EX': 'Existential there', 'FW': 'Foreign word', 'IN':
'Preposition/subordinating conjunction',
    'JJ': 'Adjective', 'JJR': 'Adjective, comparative', 'JJS': 'Adjective,
superlative',
    'LS': 'List item marker', 'MD': 'Modal', 'NN': 'Noun, singular or mass',
'NNS': 'Noun, plural',
    'NNP': 'Proper noun, singular', 'NNPS': 'Proper noun, plural', 'PDT':
'Predeterminer',
    'POS': 'Possessive ending', 'PRP': 'Personal pronoun', 'PRP$':
'Possessive pronoun',
    'RB': 'Adverb', 'RBR': 'Adverb, comparative', 'RBS': 'Adverb,
superlative', 'RP': 'Particle',
    'TO': 'To', 'UH': 'Interjection', 'VB': 'Verb, base form', 'VBD': 'Verb,
past tense',
    'VBG': 'Verb, gerund or present participle', 'VBN': 'Verb, past
participle',
    'VBP': 'Verb, non-3rd person singular present', 'VBZ': 'Verb, 3rd person
singular present',
    'WDT': 'Wh-determiner', 'WP': 'Wh-pronoun', 'WP$': 'Possessive
wh-pronoun', 'WRB': 'Wh-adverb'
}
def wordnet_pos_code(tag):
    if tag.startswith('J'):
        return wn.ADJ
    elif tag.startswith('V'):
        return wn.VERB
```

```python
    elif tag.startswith('N'):
        return wn.NOUN
    elif tag.startswith('R'):
        return wn.ADV
    else:
        return None

tokenization = tokenized_texts

total_treebank_time = 0
total_wordnet_time = 0
total_precision = 0
total_sentences = len(tokenization)

def get_pos_full_form(tag):
    return pos_full_form.get(tag, "Unknown")

for tokens in tokenization:
    print(f"\nProcessing sentence: {' '.join(tokens)}")
    word = tokens[0]

    synsets = wn.synsets(word)
    print(f"\nSynsets of '{word}':")
    for synset in synsets:
        print(f"- {synset.name()}: {synset.definition()}")
    print(f"\nLemmas of '{word}':")
    for synset in synsets:
        lemmas = synset.lemmas()
        for lemma in lemmas:
            print(f"- {lemma.name()}")
    print(f"\nHypernyms of '{word}':")
    for synset in synsets:
        hypernyms = synset.hypernyms()
        for hypernym in hypernyms:
            print(f"- {synset.name()} -> {hypernym.name()}")
    print(f"\nHyponyms of '{word}':")
    for synset in synsets:
        hyponyms = synset.hyponyms()
        for hyponym in hyponyms:
            print(f"- {synset.name()} -> {hyponym.name()}")
    print(f"\nExample sentences for '{word}':")
    for synset in synsets:
        examples = synset.examples()
        for example in examples:
            print(f"- {synset.name()}: {example}")
    # Treebank
    start_treebank = time()
    treebank_tags = pos_tag(tokens)
    end_treebank = time()
    print("Treebank POS tagging results:")
    for token, tag in treebank_tags:
        print(f'{token} -> {tag} ({get_pos_full_form(tag)})')

    # WordNet
    start_wordnet = time()
    wordnet_tags = []
    for token, tag in treebank_tags:
        wn_tag = wordnet_pos_code(tag)
        if wn_tag:
            synsets = wn.synsets(token, wn_tag)
            if synsets:
                wn_tagged = synsets[0].pos()
            else:
                wn_tagged = "unknown"
        else:
            wn_tagged = "unknown"
        wordnet_tags.append((token, wn_tagged))
    end_wordnet = time()

    print("\nWordNet-based tagging results:")
    for token, tag in wordnet_tags:
        print(f'{token} -> {tag}')
    treebank_time = end_treebank - start_treebank
    wordnet_time = end_wordnet - start_wordnet
    total_treebank_time += treebank_time
    total_wordnet_time += wordnet_time

    correct_matches = sum(1 for t1, t2 in zip(treebank_tags, wordnet_tags) if
t1[1].startswith(t2[1].upper()))
    precision = correct_matches / len(treebank_tags)
    total_precision += precision

avg_precision = total_precision / total_sentences
avg_treebank_time = total_treebank_time / total_sentences
avg_wordnet_time = total_wordnet_time / total_sentences
print(f"\nOverall average time for Treebank POS tagging:
```

```python
{avg_treebank_time:.6f} seconds")
print(f"Overall average time for WordNet-based tagging:
{avg_wordnet_time:.6f} seconds")
print(f"Overall precision (accuracy): {avg_precision:.2f}")
```

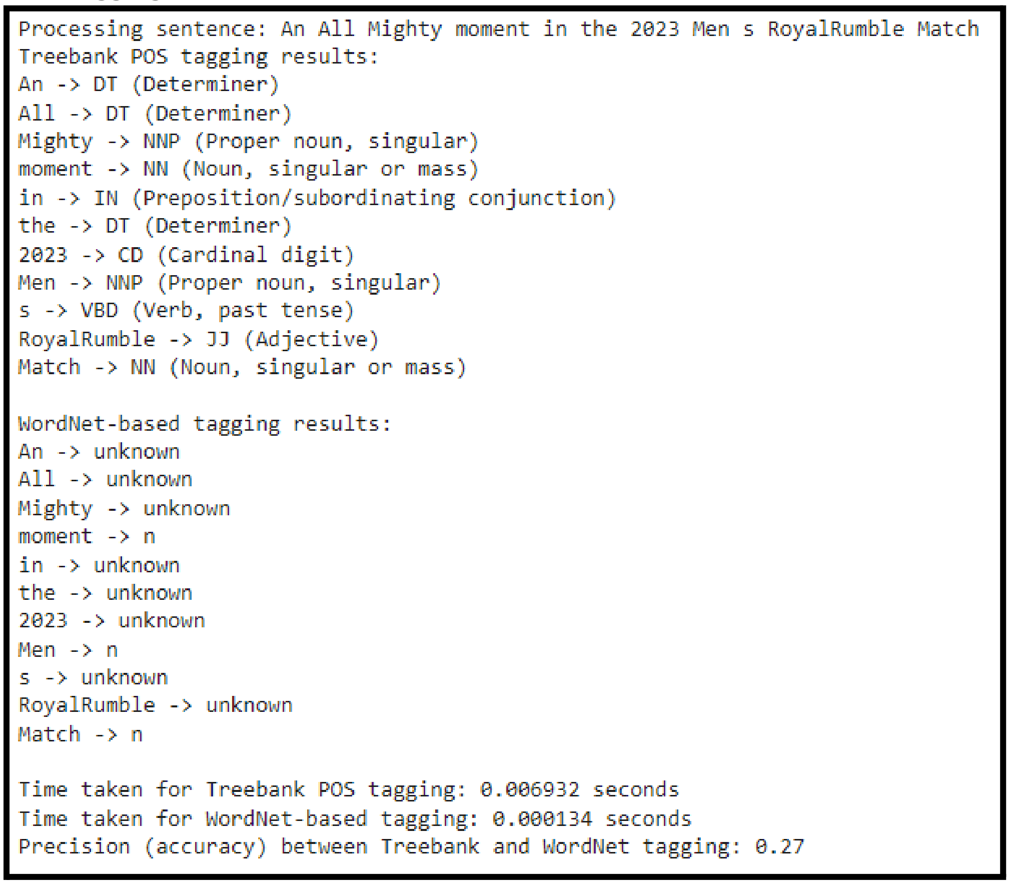
<u>Output:</u>

Wordnet library functions

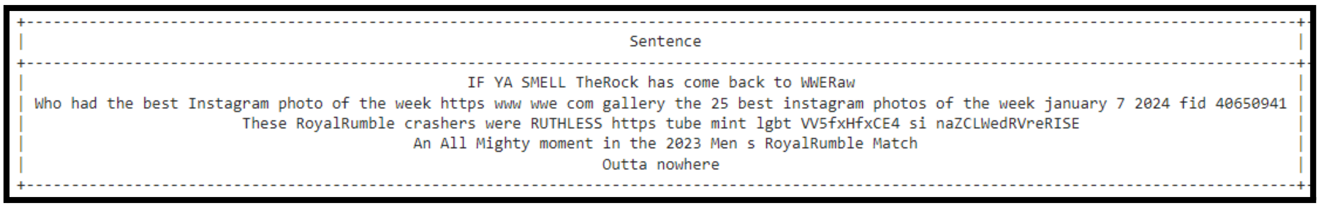


```
WordNet Exploration:

Synsets of 'Who':
- world_health_organization.n.01: a United Nations agency to coordinate international health activities and to help governments improve health services

Lemmas of 'Who':
- World_Health_Organization
- WHO

Hypernyms of 'Who':
- world_health_organization.n.01 -> united_nations_agency.n.01

Hyponyms of 'Who':

Example sentences for 'Who':
POS Tagging:
These -> DT
RoyalRumble -> JJ
crashers -> NNS
were -> VBD
RUTHLESS -> NNP
https -> NN
tube -> NN
mint -> NN
lgbt -> NN
VV5fxHfxCE4 -> NNP
si -> NN
naZCLWedRVreRISE -> NN
```

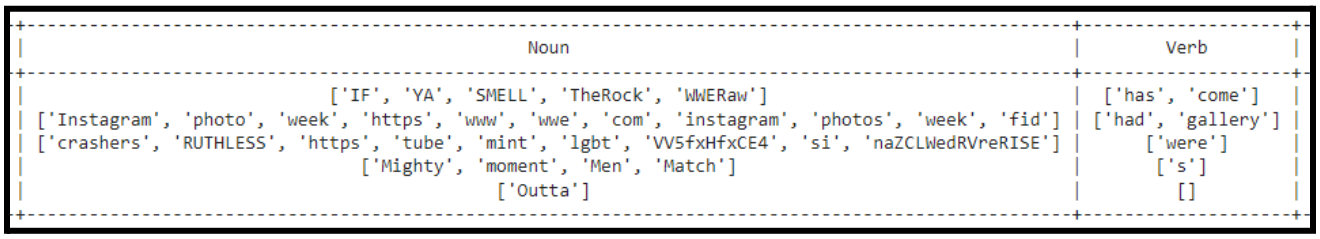
POS Tagging



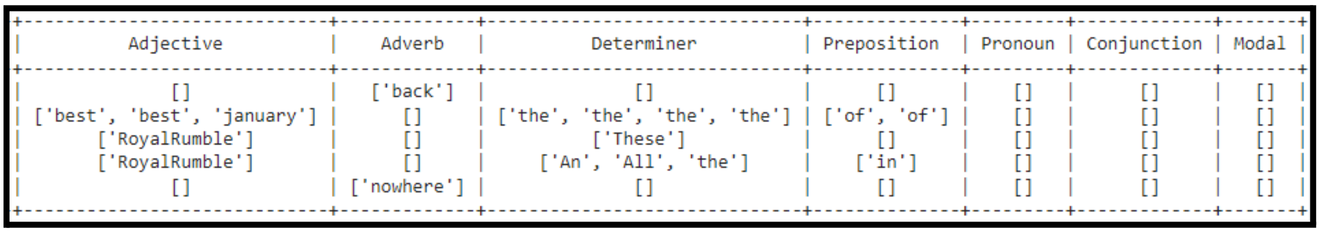
```
Processing sentence: An All Mighty moment in the 2023 Men s RoyalRumble Match
Treebank POS tagging results:
An -> DT (Determiner)
All -> DT (Determiner)
Mighty -> NNP (Proper noun, singular)
moment -> NN (Noun, singular or mass)
in -> IN (Preposition/subordinating conjunction)
the -> DT (Determiner)
2023 -> CD (Cardinal digit)
Men -> NNP (Proper noun, singular)
s -> VBD (Verb, past tense)
RoyalRumble -> JJ (Adjective)
Match -> NN (Noun, singular or mass)

WordNet-based tagging results:
An -> unknown
All -> unknown
Mighty -> unknown
moment -> n
in -> unknown
the -> unknown
2023 -> unknown
Men -> n
s -> unknown
RoyalRumble -> unknown
Match -> n

Time taken for Treebank POS tagging: 0.006932 seconds
Time taken for WordNet-based tagging: 0.000134 seconds
Precision (accuracy) between Treebank and WordNet tagging: 0.27
```



```
+--------------------------------------------------------------------------------------------------------------------------+
|                                                         Sentence                                                         |
+--------------------------------------------------------------------------------------------------------------------------+
|                                     IF YA SMELL TheRock has come back to WWERaw                                          |
| Who had the best Instagram photo of the week https www wwe com gallery the 25 best instagram photos of the week january 7 2024 fid 40650941 |
|                 These RoyalRumble crashers were RUTHLESS https tube mint lgbt VV5fxHfxCE4 si naZCLWedRVreRISE            |
|                                 An All Mighty moment in the 2023 Men s RoyalRumble Match                                 |
|                                                       Outta nowhere                                                      |
+--------------------------------------------------------------------------------------------------------------------------+
```



```
+-----------------------------------------------------------------------------------------------------+-------------------+
|                                                Noun                                                 |        Verb       |
+-----------------------------------------------------------------------------------------------------+-------------------+
|                            ['IF', 'YA', 'SMELL', 'TheRock', 'WWERaw']                               |  ['has', 'come']  |
| ['Instagram', 'photo', 'week', 'https', 'www', 'wwe', 'com', 'instagram', 'photos', 'week', 'fid']  | ['had', 'gallery'] |
| ['crashers', 'RUTHLESS', 'https', 'tube', 'mint', 'lgbt', 'VV5fxHfxCE4', 'si', 'naZCLWedRVreRISE']  |      ['were']     |
|                            ['Mighty', 'moment', 'Men', 'Match']                                     |       ['s']       |
|                                           ['Outta']                                                 |         []        |
+-----------------------------------------------------------------------------------------------------+-------------------+
```



```
+------------------------------+--------------+------------------------------+-------------+---------+-------------+-------+
|          Adjective           |    Adverb    |          Determiner          | Preposition | Pronoun | Conjunction | Modal |
+------------------------------+--------------+------------------------------+-------------+---------+-------------+-------+
|              []              |   ['back']   |              []              |      []     |    []   |      []     |   []  |
| ['best', 'best', 'january']  |      []      | ['the', 'the', 'the', 'the'] | ['of', 'of'] |    []   |      []     |   []  |
|        ['RoyalRumble']       |      []      |          ['These']           |      []     |    []   |      []     |   []  |
|        ['RoyalRumble']       |      []      |     ['An', 'All', 'the']     |    ['in']   |    []   |      []     |   []  |
|              []              | ['nowhere']  |              []              |      []     |    []   |      []     |   []  |
+------------------------------+--------------+------------------------------+-------------+---------+-------------+-------+
```

Conclusion:

- Utilizes NLTK for tagging parts of speech in tokenized sentences.
- Maps POS tags to categories like Noun, Verb, and Adjective, Determiner, Preposition, Pronoun, Conjunction, and Modal.
- Measures and compares the time taken for Treebank and WordNet tagging.
- Provides a structured way to analyze and visualize POS tagging results.