Experiment No.7
Implement Named Entity Recognizer for the given Text input
Date of Performance:
Date of Submission:

Aim: Implement Named Entity Recognizer for the given Text input

Objective: Understand the importance of NER in NLP and Implement NER.

Theory:

The named entity recognition (NER) is one of the most data preprocessing task. It involves the identification of key information in the text and classification into a set of predefined categories. An entity is basically the thing that is consistently talked about or refer to in the text.

NER is the form of NLP.

At its core, NLP is just a two-step process, below are the two steps that are involved:

- Detecting the entities from the text
- Classifying them into different categories

Some of the categories that are the most important architecture in NER such that:

- Person
- Organization
- Place/ location

Other common tasks include classifying of the following:

- date/time.
- expression
- Numeral measurement (money, percent, weight, etc)
- E-mail address

Ambiguity in NE

For a person, the category definition is intuitively quite clear, but for computers, there is some ambiguity in classification. Let's look at some ambiguous example:

England (Organisation) won the 2019 world cup vs The 2019 world cup happened in England(Location).

Washington(Location) is the capital of the US vs The first president of the US was Washington(Person).

```
import spacy
```

Load the spaCy language model

```
nlp = spacy.load("en_core_web_sm")
```

Sample text input

```
text = "Apple Inc. is a company based in Cupertino, California. John works for Google in Mountain View."
```

Process the text using spaCy

```
import spacy

# Load the spaCy language model
nlp = spacy.load("en_core_web_sm")

# Sample text input
text = "Apple Inc. is a company based in Cupertino, California. John works for Google in Mountain View."

# Process the text using spaCy
doc = nlp(text)

# Initialize variables to store named entities
named_entities = []

# Define a function to extract named entities
def extract_named_entities(doc):
    entities = []
    current_entity = None

    for token in doc:
        if token.ent_type_:
            if current_entity and token.ent_type_ == current_entity[1]:
                current_entity = (current_entity[0] + " " + token.text, token.ent_type_)
            else:
                if current_entity:
                    entities.append(current_entity)
                current_entity = (token.text, token.ent_type_)
        else:
            if current_entity:
                entities.append(current_entity)
            current_entity = None

    if current_entity:
        entities.append(current_entity)

    return entities

# Extract named entities
named_entities = extract_named_entities(doc)

# Print the named entities
for entity, label in named_entities:
    print(f"Entity: {entity}, Label: {label}")
```

```
Entity: Apple Inc., Label: ORG
Entity: Cupertino, Label: GPE
Entity: California, Label: GPE
Entity: John, Label: PERSON
Entity: Google, Label: ORG
Entity: Mountain View, Label: GPE
```

Conclusion:

The results appear to be mostly accurate in identifying named entities and their corresponding labels. "Apple Inc." is correctly labeled as an organization (ORG), "Cupertino" and "California" are accurately tagged as geographical locations (GPE), "John" is correctly identified as a person (PERSON), and "Google" is appropriately labeled as an organization (ORG). "Mountain View" is also correctly recognized as a geographical location (GPE). In conclusion, the named entity recognition results provided seem to be accurate. However, it's important to note that the correctness of these results can be context-dependent. The accuracy of named entity recognition can vary based on the specific task or dataset, and it's essential to ensure that the model's performance is consistent and reliable in the intended application.