**Tokenization and Part-of-Speech Tagging**

import spacy

# Load the English language model

nlp = spacy.load('en\_core\_web\_sm')

# Load the docx2txt package

import docx2txt

# Replace 'path\_to\_your\_document.docx' with the path to your Word document

text = docx2txt.process(r"C:\Users\reply\Desktop\disease.docx")

# Process the text with spaCy

doc = nlp(text)

# Access tokens and POS tags

tokens\_and\_pos = [(token.text, token.pos\_) for token in doc]

# Print tokens and POS tags

for token, pos in tokens\_and\_pos:

print(f"Token: {token}, POS: {pos}")

print("\n")

# Extract entities and their labels

entities = [(ent.text, ent.label\_) for ent in doc.ents]

# Print entities and their labels

for entity, label in entities:

print(f"Entity: {entity}, Label: {label}")

print("\n")

# Dependency parsing

print("Dependency parsing:")

for token in doc:

print(f"{token.text} --> {token.dep\_} --> {token.head.text}")

# Chunking (noun phrases and verb phrases)

print("\nChunking:")

for chunk in doc.noun\_chunks:

print(f"Noun chunk: {chunk.text}")

for token in doc:

if token.pos\_ == "VERB":

print(f"Verb chunk: {token.text}")

print("\n")

def extract\_features(text):

doc = nlp(text)

features = []

for token in doc:

if token.pos\_ in ['NOUN', 'VERB']: # Consider nouns and verbs as features

features.append(token.text.lower()) # Store lowercase text as feature

return features

# Example text

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

features = extract\_features(text)

print("Extracted Features:", features)

print("\n")

# Process text with spaCy

doc = nlp(text)

# Get word vectors from spaCy (using GloVe embeddings)

word\_vectors = [token.vector for token in doc if token.has\_vector]

print("Word Vectors:")

for vector in word\_vectors:

print(vector)

**Named Entity Recognition**

Performing Named Entity Recognition (NER) to identify and classify entities such as diseases, symptoms, medications, and demographics within text using spaCy involves leveraging pre-trained models or training custom models tailored to the medical domain. Here’s how you can approach it using spaCy: