National University
Of Computer and Emerging Sciences

Name:

Dawood Sarfraz

Roll no:

20P-0153

Section:

BSCS-7A

Course:

Natural Language Processing

Lab:

03

Submitted to:

Dr. Omer Usman Khan

FAST National University of Computer and Emerging Sciences

Task 1:

1. What is en_core_web_sm?

In natural language processing (NLP), **`en_core_web_sm`** refers to a specific English language model provided by the **spaCy** library. The **"en"** signifies English, **"core"** suggests that it's a core or base model, **"web"** implies it has been trained on a mixture of web text for versatility, and **"sm"** denotes that it is a small model. This model provides **pre-trained** capabilities for tasks such as **part-of-speech tagging**, **named entity recognition**, and **dependency parsing**. It strikes a balance between model size and performance, making it a practical choice for various NLP applications.

Tokenizer:
- Function: Breaks down a given text into individual words, punctuations, and other meaningful units called tokens.
- Example: "The quick brown fox" would be tokenized into ["The", "quick", "brown", "fox"].

Tagger:
- Function: Assigns parts-of-speech (POS) tags to each token, indicating the grammatical category of the word.
- Example: Tagging "dog" as a noun (NN) and "run" as a verb (VB).

Parser:
- Function: Analyzes the grammatical structure of sentences, determining how words relate to each other syntactically.
- Example: Identifying subject-verb-object relationships in a sentence.

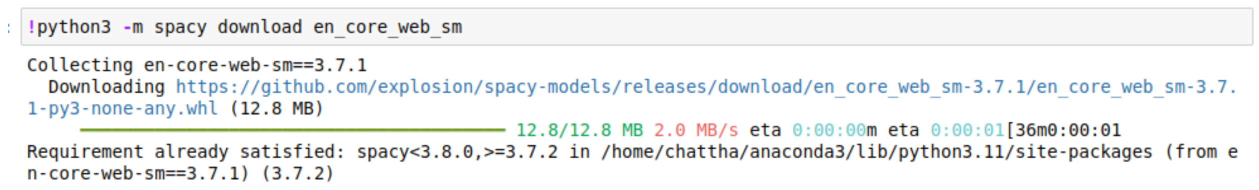
Named Entity Recognition (NER):
- Function: Identifies and classifies named entities (e.g., persons, organizations, locations) within the text.
- Example: Recognizing "Apple" as an organization and "New York" as a location.

Attribute Ruler:
- Function: Applies custom rules to extract additional information or attributes from the text.
- Example: Extracting dates, quantities, or custom patterns based on predefined rules.

Lemmatizer:
- Function: Reduces words to their base or root form (lemma), simplifying variations of a word to a common base.
- Example: Lemmatizing "running" to "run" or "better" to "good."

2. What is the size of en_core_web_sm?

12.8 MB



```
!python3 -m spacy download en_core_web_sm
```

```
Collecting en-core-web-sm==3.7.1
  Downloading https://github.com/explosion/spacy-models/releases/download/en_core_web_sm-3.7.1/en_core_web_sm-3.7.
1-py3-none-any.whl (12.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 12.8/12.8 MB 2.0 MB/s eta 0:00:00m eta 0:00:01[36m0:00:01
Requirement already satisfied: spacy<3.8.0,>=3.7.2 in /home/chattha/anaconda3/lib/python3.11/site-packages (from e
n-core-web-sm==3.7.1) (3.7.2)
```

3. What other variations can be used?

"en_core_web_md "(Medium):

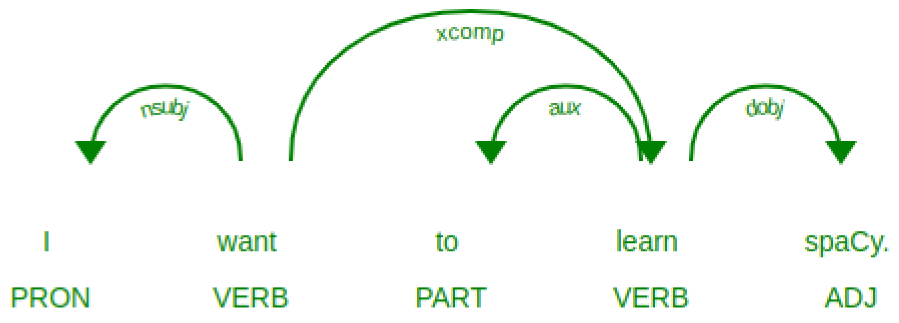
- Description: This is a medium-sized English model in spaCy.
- Features: It includes more vectors for word representations, making it more suitable for tasks requiring a richer understanding of word meanings.
- Use Cases: It's a good choice when more detailed word embeddings are needed, and the computational resources allow for a larger model.

"en_core_web_lg" (Large):

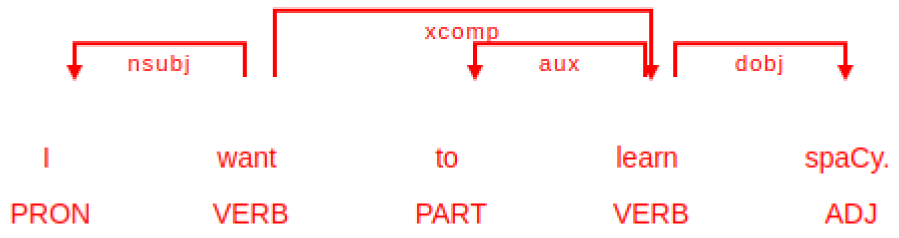
- Description: This is the large English model in spaCy.
- Features: It includes even more vectors for word representations, providing a more extensive and detailed language understanding.
- Use Cases: Suitable for tasks demanding a high level of accuracy and semantic understanding. It's a larger model, so it requires more computational resources.

<u>Task 2:</u>

1. ***Draw the left and right dependencies for the sentence: I want to learn spaCy.***

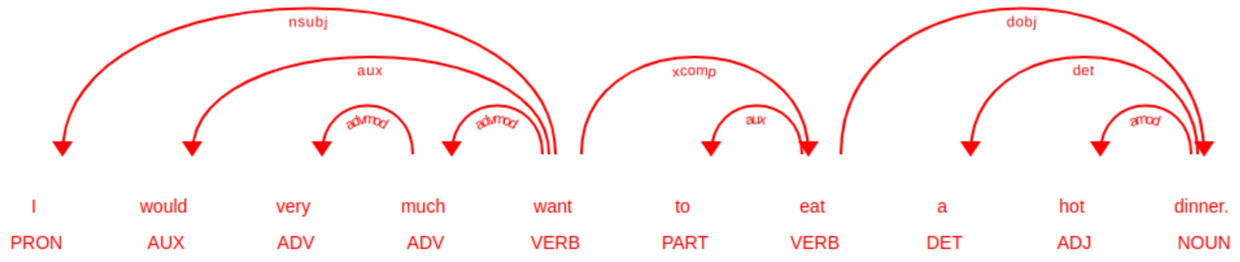


2. ***Draw the children for the sentence: I want to learn spaCy.***

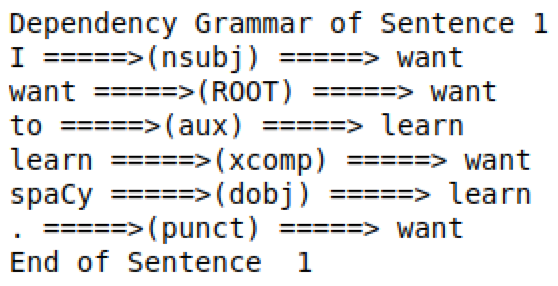


—

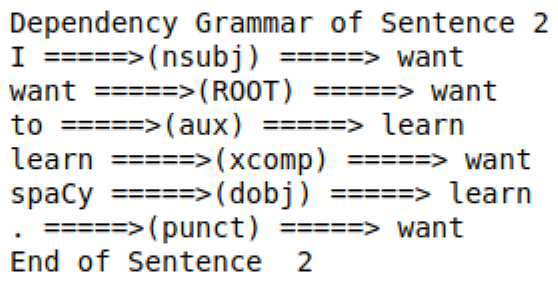
3. ***Draw the left and right dependencies for the sentence: I would very much want to eat a hot dinner.***



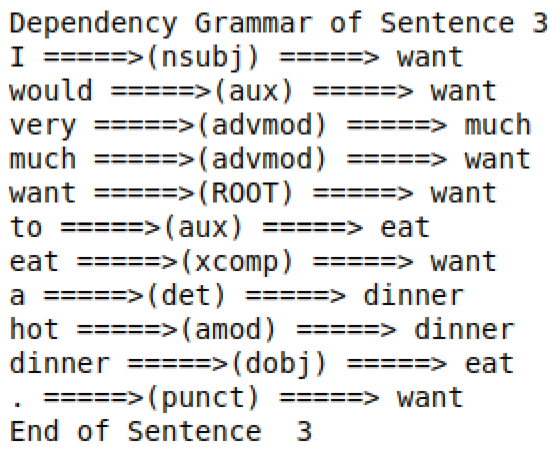
4. ***Present a list of all dependency grammars of your sentences above.***



```
Dependency Grammar of Sentence 1
I =====>(nsubj) =====> want
want =====>(ROOT) =====> want
to =====>(aux) =====> learn
learn =====>(xcomp) =====> want
spaCy =====>(dobj) =====> learn
. =====>(punct) =====> want
End of Sentence  1
```

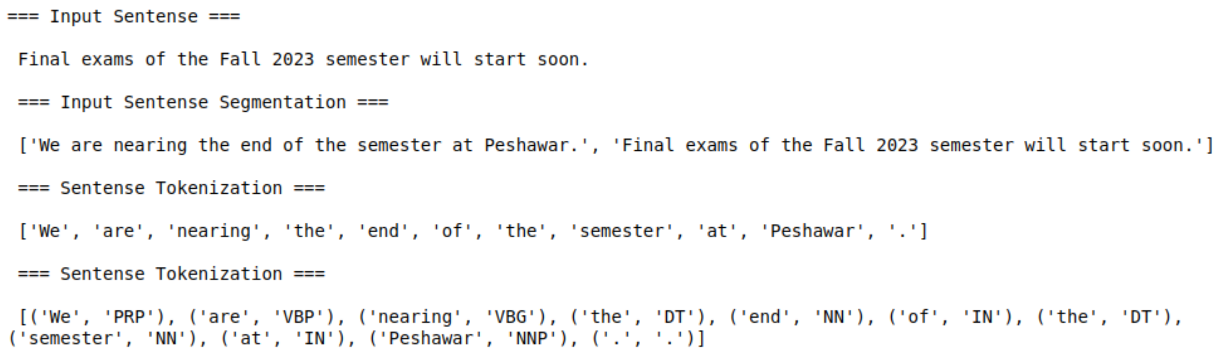


```
Dependency Grammar of Sentence 2
I =====>(nsubj) =====> want
want =====>(ROOT) =====> want
to =====>(aux) =====> learn
learn =====>(xcomp) =====> want
spaCy =====>(dobj) =====> learn
. =====>(punct) =====> want
End of Sentence  2
```

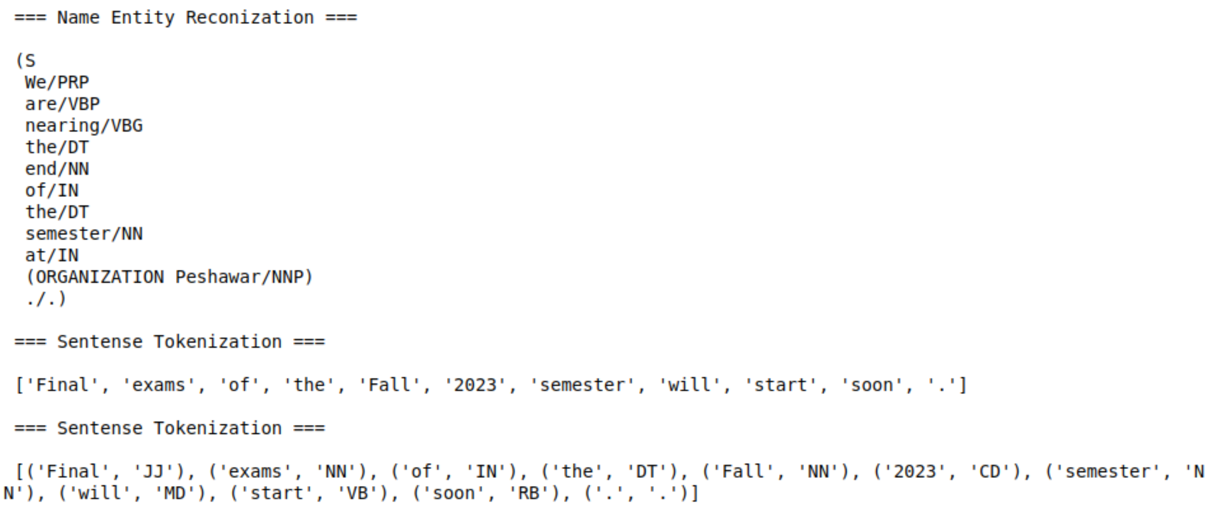


```
Dependency Grammar of Sentence 3
I =====>(nsubj) =====> want
would =====>(aux) =====> want
very =====>(advmod) =====> much
much =====>(advmod) =====> want
want =====>(ROOT) =====> want
to =====>(aux) =====> eat
eat =====>(xcomp) =====> want
a =====>(det) =====> dinner
hot =====>(amod) =====> dinner
dinner =====>(dobj) =====> eat
. =====>(punct) =====> want
End of Sentence  3
```

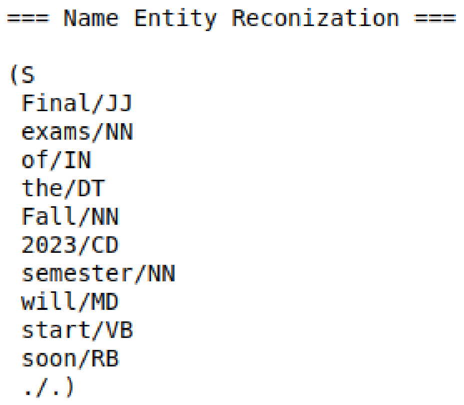
1. How did the Named Entity Output of the NLTK pipeline look like? Present its output.



```
=== Input Sentense ===

 Final exams of the Fall 2023 semester will start soon.

 === Input Sentense Segmentation ===

 ['We are nearing the end of the semester at Peshawar.', 'Final exams of the Fall 2023 semester will start soon.']

 === Sentense Tokenization ===

 ['We', 'are', 'nearing', 'the', 'end', 'of', 'the', 'semester', 'at', 'Peshawar', '.']

 === Sentense Tokenization ===

 [('We', 'PRP'), ('are', 'VBP'), ('nearing', 'VBG'), ('the', 'DT'), ('end', 'NN'), ('of', 'IN'), ('the', 'DT'),
('semester', 'NN'), ('at', 'IN'), ('Peshawar', 'NNP'), ('.', '.')]
```

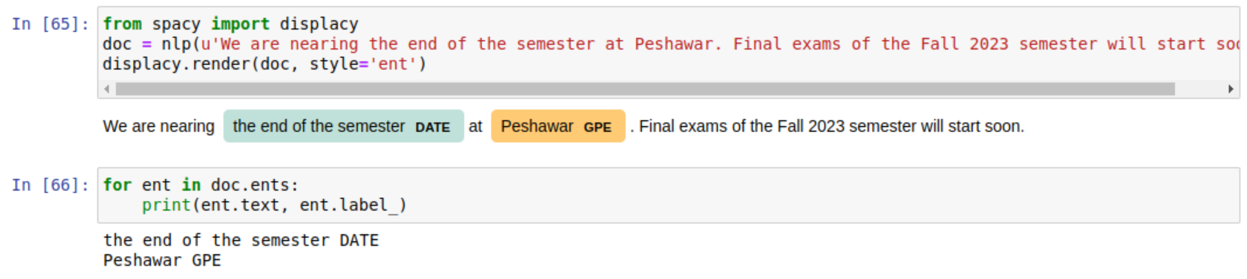


```
 === Name Entity Reconization ===

 (S
  We/PRP
  are/VBP
  nearing/VBG
  the/DT
  end/NN
  of/IN
  the/DT
  semester/NN
  at/IN
  (ORGANIZATION Peshawar/NNP)
  ./.)

 === Sentense Tokenization ===

 ['Final', 'exams', 'of', 'the', 'Fall', '2023', 'semester', 'will', 'start', 'soon', '.']

 === Sentense Tokenization ===

 [('Final', 'JJ'), ('exams', 'NN'), ('of', 'IN'), ('the', 'DT'), ('Fall', 'NN'), ('2023', 'CD'), ('semester', 'N
N'), ('will', 'MD'), ('start', 'VB'), ('soon', 'RB'), ('.', '.')]
```



```
 === Name Entity Reconization ===

 (S
  Final/JJ
  exams/NN
  of/IN
  the/DT
  Fall/NN
  2023/CD
  semester/NN
  will/MD
  start/VB
  soon/RB
  ./.)
```

1. *How did the Named Entity Output of the spaCy pipeline look like? Present its output.*

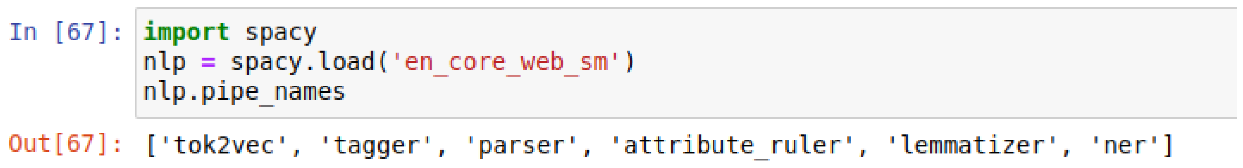


```
In [65]: from spacy import displacy
         doc = nlp(u'We are nearing the end of the semester at Peshawar. Final exams of the Fall 2023 semester will start so
         displacy.render(doc, style='ent')
```

We are nearing the end of the semester DATE at Peshawar GPE . Final exams of the Fall 2023 semester will start soon.

```
In [66]: for ent in doc.ents:
             print(ent.text, ent.label_)

the end of the semester DATE
Peshawar GPE
```

1. *What is the default pipeline structure of spaCy?*



```
In [67]: import spacy
         nlp = spacy.load('en_core_web_sm')
         nlp.pipe_names

Out[67]: ['tok2vec', 'tagger', 'parser', 'attribute_ruler', 'lemmatizer', 'ner']
```