```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('punkt_tab') # Added to resolve LookupError
sentence = "Students are learning Natural Language Processing"
tokens = nltk.word_tokenize(sentence)
pos_tags = nltk.pos_tag(tokens)
print(pos_tags)
```

```
o /root/nltk_data...
up-to-date!
d_perceptron_tagger_eng to

on_tagger_eng is already up-to-

ab to /root/nltk_data...
t_tab.zip.
'learning', 'VBG'), ('Natural', 'NNP'), ('Language', 'NNP'), ('Processing'
```

```
import spacy
nlp = spacy.load("en_core_web_sm")
doc = nlp("Students are learning Natural Language Processing")
for token in doc:
    print(token.text, token.pos_)
```

```
Students NOUN
are AUX
learning VERB
Natural PROPN
Language PROPN
Processing NOUN
```

```
nlp = spacy.load("en_core_web_sm")
doc = nlp("Apple is looking at buying a startup in India.")
for token in doc:
    print(token.text, token.pos_, token.tag_)
```

```
Apple PROPN NNP
is AUX VBZ
looking VERB VBG
at ADP IN
buying VERB VBG
a DET DT
startup NOUN NN
in ADP IN
India PROPN NNP
. PUNCT .
```

```
import spacy
from collections import Counter
nlp = spacy.load("en_core_web_sm")
text = "Loving the new AI features 😍 #AI #MachineLearning"
doc = nlp(text)
nouns = []
verbs = []
for token in doc:
    if token.pos_ in ["NOUN", "PROPN"]:
        nouns.append(token.text)
    elif token.pos_ == "VERB":
        verbs.append(token.text)
noun_freq = Counter(nouns)
verb_freq = Counter(verbs)
print("Noun Frequency:", noun_freq)
print("Verb Frequency:", verb_freq)
```

```
Noun Frequency: Counter({'AI': 2, '😍': 1, 'MachineLearning': 1})
Verb Frequency: Counter({'Loving': 1, 'features': 1})
```

```
SRUniversity="""The SR University campus is located in Ananthasagar vill
It is in 150 acres, with both separate hostel facilities for boys and gi
There is a huge central library along with Indias largest Technology Bus
```

```
nltk.download('averaged_perceptron_tagger')
from nltk.tokenize import word_tokenize
words = word_tokenize(SRUniversity)
nltk.pos_tag(words)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
[('The', 'DT'),
 ('SR', 'NNP'),
 ('University', 'NNP'),
 ('campus', 'NN'),
 ('is', 'VBZ'),
 ('located', 'VBN'),
 ('in', 'IN'),
 ('Ananthasagar', 'NNP'),
 ('village', 'NN'),
 ('of', 'IN'),
 ('Hasanparthy', 'NNP'),
 ('Mandal', 'NNP'),
 ('in', 'IN'),
 ('Warangal', 'NNP'),
 (',', ','),
 ('Telangana', 'NNP'),
 (',', ','),
 ('India', 'NNP'),
```

```
 ('.', '.'),
 ('It', 'PRP'),
 ('is', 'VBZ'),
 ('in', 'IN'),
 ('150', 'CD'),
 ('acres', 'NNS'),
 (',', ','),
 ('with', 'IN'),
 ('both', 'DT'),
 ('separate', 'JJ'),
 ('hostel', 'NN'),
 ('facilities', 'NNS'),
 ('for', 'IN'),
 ('boys', 'NNS'),
 ('and', 'CC'),
 ('girls', 'NNS'),
 ('.', '.'),
 ('There', 'EX'),
 ('is', 'VBZ'),
 ('a', 'DT'),
 ('huge', 'JJ'),
 ('central', 'JJ'),
 ('library', 'NN'),
 ('along', 'IN'),
 ('with', 'IN'),
 ('Indias', 'NNP'),
 ('largest', 'JJS'),
 ('Technology', 'NN'),
 ('Business', 'NNP'),
 ('Incubator', 'NNP'),
 ('(', '('),
 ('TBI', 'NNP'),
 (')', ')'),
 ('in', 'IN'),
 ('tier', '$'),
 ('2', 'CD'),
 ('cities', 'NNS')
```

```
nltk.download('tagsets')
nltk.download('tagsets_json') # Added to download the necessary resource
nltk.help.upenn_tagset()
```

```
dollar
 $ -$ --$ A$ C$ HK$ M$ NZ$ S$ U.S.$ US$
 closing quotation mark
 ' ''
opening parenthesis
 ( [ {
closing parenthesis
 ) ] }
comma
 ,
 dash
 --
sentence terminator
 . ! ?
```

```
colon or ellipsis
 : ; ...
 conjunction, coordinating
 & 'n and both but either et for less minus neither nor or plus so
 therefore times v. versus vs. whether yet
 numeral, cardinal
 mid-1890 nine-thirty forty-two one-tenth ten million 0.5 one forty-
 seven 1987 twenty '79 zero two 78-degrees eighty-four IX '60s .025
 fifteen 271,124 dozen quintillion DM2,000 ...
 determiner
 all an another any both del each either every half la many much nary
 neither no some such that the them these this those
 existential there
 there
 foreign word
 gemeinschaft hund ich jeux habeas Haementeria Herr K'ang-si vous
 lutihaw alai je jour objets salutaris fille quibusdam pas trop Monte
 terram fiche oui corporis ...
 preposition or conjunction, subordinating
 astride among uppon whether out inside pro despite on by throughout
 below within for towards near behind atop around if like until below
 next into if beside ...
 adjective or numeral, ordinal
 third ill-mannered pre-war regrettable oiled calamitous first separable
 ectoplasmic battery-powered participatory fourth still-to-be-named
 multilingual multi-disciplinary ...
: adjective, comparative
 bleaker braver breezier briefer brighter brisker broader bumper busier
 calmer cheaper choosier cleaner clearer closer colder commoner costlier
 cozier creamier crunchier cuter ...
: adjective, superlative
 calmest cheapest choicest classiest cleanest clearest closest commonest
 corniest costliest crassest creepiest crudest cutest darkest deadliest
 dearest deepest densest dinkiest ...
 list item marker
 A A. B B. C C. D E F First G H I J K One SP-44001 SP-44002 SP-44005
 SP-44007 Second Third Three Two * a b c d first five four one six three
 two
 modal auxiliary
 can cannot could couldn't dare may might must need ought shall should
 shouldn't will would
 noun, common, singular or mass
 common-carrier cabbage knuckle-duster Casino afghan shed thermostat
```

```
# Install libraries
!pip install nltk spacy
!python -m spacy download en_core_web_sm
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.12/dist-pac
Requirement already satisfied: spacy in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: click in /usr/local/lib/python3.12/dist-pa
Requirement already satisfied: joblib in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.
Requirement already satisfied: tqdm in /usr/local/lib/python3.12/dist-pac
```

```
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/li
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/pyth
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/py
Requirement already satisfied: thinc<8.4.0,>=8.3.4 in /usr/local/lib/pyth
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/pyt
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/pyth
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/
Requirement already satisfied: weasel<0.5.0,>=0.4.2 in /usr/local/lib/pyt
Requirement already satisfied: typer-slim<1.0.0,>=0.3.0 in /usr/local/lib
Requirement already satisfied: numpy>=1.19.0 in /usr/local/lib/python3.12
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /u
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-p
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/di
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/p
Requirement already satisfied: pydantic-core==2.41.4 in /usr/local/lib/py
Requirement already satisfied: typing-extensions>=4.14.1 in /usr/local/li
Requirement already satisfied: typing-inspection>=0.4.2 in /usr/local/lib
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/pytho
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/pytho
Requirement already satisfied: blis<1.4.0,>=1.3.0 in /usr/local/lib/pytho
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib
Requirement already satisfied: cloudpathlib<1.0.0,>=0.7.0 in /usr/local/l
Requirement already satisfied: smart-open<8.0.0,>=5.2.1 in /usr/local/lib
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.
Requirement already satisfied: wrapt in /usr/local/lib/python3.12/dist-pa
Collecting en-core-web-sm==3.8.0
  Downloading https://github.com/explosion/spacy-models/releases/download
```

```
✔ Download and installation successful
You can now load the package via spacy.load('en_core_web_sm')
⚠ Restart to reload dependencies
If you are in a Jupyter or Colab notebook, you may need to restart Python
order to load all the package's dependencies. You can do this by selectin
'Restart kernel' or 'Restart runtime' option.
```

```
import nltk
import spacy
import re
from collections import Counter

nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('twitter_samples')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
```

```
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package twitter_samples to /root/nltk_data...
[nltk_data]   Unzipping corpora/twitter_samples.zip.
True
```

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

```
file_path = "/content/enriched_posts.txt"

with open(file_path, "r", encoding="utf-8") as f:
    text = f.read()

print(text[:500])  # preview first 500 characters
```

```
[
    {
        "text": "Just saw a LinkedIn Influencer with 'Organic Growth' wri
        "engagement": 90,
        "line_count": 3,
        "language": "English",
        "tags": [
            "Influencer",
            "Personal Growth"
        ],
        "tone": "Professional"
    },
    {
        "text": "Jobseekers, this one\u2019s for you.\n Every applic
```

```
def preprocess(text):
    text = re.sub(r"http\S+", "", text)   # remove URLs
    text = re.sub(r"@\w+", "", text)      # remove mentions
    return text

clean_text = preprocess(text)

# Tokenize using NLTK
tokens_nltk = nltk.word_tokenize(clean_text)

# Tokenize using spaCy
doc = nlp(clean_text)
tokens_spacy = [token.text for token in doc]

print("NLTK Tokens:", tokens_nltk[:20])
print("spaCy Tokens:", tokens_spacy[:20])
```

```
NLTK Tokens: ['[', '{', '``', 'text', "'", ':', '``', 'Just', 'saw', 'a'
spaCy Tokens: ['[', '\n    ', '{', '\n        ', '"', 'text', '"', ':', '
```

```
pos_tags_nltk = nltk.pos_tag(tokens_nltk)

pos_tags_nltk[:20]
```

```
[('[', 'RB'),
 ('{', '('),
 ('``', '``'),
 ('text', 'NN'),
 ("'", "'"),
 (':', ':'),
 ('``', '``'),
 ('Just', 'RB'),
 ('saw', 'VB'),
 ('a', 'DT'),
 ('LinkedIn', 'NNP'),
 ('Influencer', 'NNP'),
 ('with', 'IN'),
 ("'Organic", 'JJ'),
 ('Growth', 'NNP'),
 ("'", 'POS'),
 ('written', 'VBN'),
 ('in', 'IN'),
 ('the', 'DT'),
 ('profile', 'NN')]
```

```
pos_tags_spacy = [(token.text, token.pos_) for token in doc]

pos_tags_spacy[:20]
```

```
[('[', 'X'),
 ('\n    ', 'SPACE'),
 ('{', 'PUNCT'),
 ('\n        ', 'SPACE'),
 ('"', 'PUNCT'),
 ('text', 'NOUN'),
 ('"', 'PUNCT'),
 (':', 'PUNCT'),
 ('"', 'PUNCT'),
 ('Just', 'ADV'),
 ('saw', 'VERB'),
 ('a', 'DET'),
 ('LinkedIn', 'NOUN'),
 ('Influencer', 'NOUN'),
 ('with', 'ADP'),
 ("'", 'PUNCT'),
 ('Organic', 'PROPN'),
 ('Growth', 'PROPN'),
```

```
 ('"', 'PUNCT'),
 ('written', 'VERB')]
```

```
slang_examples = ["lol", "omg", "idk", "🔥", "#awesome"]

for word in slang_examples:
    print("Word:", word)
    print("NLTK:", nltk.pos_tag([word]))
    print("spaCy:", [(token.text, token.pos_) for token in nlp(word)])
    print()
```

```
Word: lol
NLTK: [('lol', 'NN')]
spaCy: [('lol', 'NOUN')]

Word: omg
NLTK: [('omg', 'NN')]
spaCy: [('omg', 'NOUN')]

Word: idk
NLTK: [('idk', 'NN')]
spaCy: [('idk', 'PROPN')]

Word: 🔥
NLTK: [('🔥', 'NN')]
spaCy: [('🔥', 'NOUN')]

Word: #awesome
NLTK: [('#awesome', 'NN')]
spaCy: [('#', 'SYM'), ('awesome', 'ADV')]
```

```
nouns = [token.text.lower() for token in doc
         if token.pos_ in ["NOUN", "PROPN"]]

noun_freq = Counter(nouns)
noun_freq.most_common(10)
```

```
[('text', 60),
 ('engagement', 60),
 ('language', 60),
 ('tags', 60),
 ('tone', 60),
 ('job', 17),
 ('t', 15),
 ('work', 11),
 ('growth', 10),
 ('hindi', 10)]
```

```
verbs = [token.lemma_.lower() for token in doc if token.pos_ == "VERB"]

verb_freq = Counter(verbs)
verb_freq.most_common(10)
```

```
[('line_count', 60),
 ('feel', 10),
 ('see', 9),
 ('know', 8),
 ('have', 8),
 ('tell', 7),
 ('work', 6),
 ('need', 6),
 ('do', 6),
 ('make', 6)]
```

```
hashtags = re.findall(r"#\w+", text)
Counter(hashtags).most_common(10)
```

```
[]
```