Parts of Speech (POS) tagging is the process of assigning a grammatical category (such as noun, verb, adjective, etc.) to each word (token) in a sentence based on its definition and context.

It answers the question: What role does each word play in this sentence?

Category	Description	Example
Noun (NN)	Name of entity	student, AI
Verb (VB)	Action or state	run, learn
Adjective (JJ)	Describes noun	smart, new
Adverb (RB)	Describes verb	quickly
Pronoun (PRP)	Replaces noun	he, they
Preposition (IN)	Shows relation	in, on
Determiner (DT)	Limits noun	the, a
Conjunction (CC)	Connects words	and, but
Interjection (UH)	Emotion	wow, omg

PoS Tagging helps

- Helps understand sentence structure
- Essential for Named Entity Recognition (NER)
- Used in information extraction, sentiment analysis, and machine translation

POS Tagging Example

Sentence: "Students are learning Natural Language Processing."

Word	POS
Students	NNS
are	VBP
learning	VBG
Natural	JJ
Language	NN
Processing	NN

NLTK uses the Penn Treebank tag set

```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger_eng')

sentence = "Students are learning Natural Language Processing"
tokens = nltk.word_tokenize(sentence)
pos_tags = nltk.pos_tag(tokens)

print(pos_tags)
```

```
[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_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
[('Students', 'NNS'), ('are', 'VBP'), ('learning', 'VBG'), ('Natural', 'NNP'), ('Language', 'NNP'), ('Processing', 'NNP')]
```

en_core_web_sm is a spaCy English language model.

The model **en_core_wen_sm** includes:

- Part-of-Speech (POS) tagging
- Lemmatization
- Named Entity Recognition (NER)
- Tokenization

It does NOT include word vectors (embeddings). For vectors, you need **en_core_web_md** or **en_core_web_lg**

spaCy uses **Universal POS tags** (simpler & semantic)

spaCy performs better for tweets, captions, and informal text.

```
import spacy
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
```

```
import spacy

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 .
```

POS Tagging for Social Media Text

Token	NLTK Tag	spaCy Tag
OMG	NNP	INTJ
lol	NN	INTJ
😂	NN	SYM
#AI	NN	PROPN

Token	Meaning	NLTK Tag	spaCy Tag
lol	laughing	NN	INTJ
omg	surprise	NNP	INTJ
rn	right now	NN	ADV
idk	I don't know	NN	VERB
lit	excellent	JJ	ADJ
😂	emotion	NN	SYM
#AI	hashtag/topic	NN	PROPN

```
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})
```