**1. What are Corpora?**

A **corpus (plural: corpora)** is a **large collection of text** used for linguistic analysis or training NLP models. It can be structured (annotated) or unstructured (raw text).

📌 **Examples of Corpora:**

* **Penn Treebank** (for syntactic structure analysis)
* **Wikipedia Dump** (for general NLP tasks)
* **Common Crawl** (for training large language models)

**2. What are Tokens?**

A **token** is a **unit of text** (word, punctuation, or subword) that results from tokenization.

📌 **Example:**  
Sentence: "I love NLP!"  
Tokens: ["I", "love", "NLP", "!"]

✅ **Types of Tokens:**

* **Word Tokens:** "love", "NLP"
* **Subword Tokens:** "NLP" → ["N", "L", "P"]
* **Character Tokens:** "NLP" → ["N", "L", "P"]

**3. What are Unigrams, Bigrams, and Trigrams?**

**N-grams** are **sequences of words** of length **N**.

📌 **Example:**  
Sentence: "I love NLP"

* **Unigram (N=1):** ["I", "love", "NLP"]
* **Bigram (N=2):** ["I love", "love NLP"]
* **Trigram (N=3):** ["I love NLP"]

✅ **Usage:**

* Used in **text prediction, machine translation, and speech recognition**.

**4. How to Generate N-Grams from Text?**

**Python Code Using NLTK:**

import nltk

from nltk.util import ngrams

text = "I love NLP"

tokens = nltk.word\_tokenize(text)

bigrams = list(ngrams(tokens, 2))

print(bigrams)

**Output:** [('I', 'love'), ('love', 'NLP')]

**5. Explain Lemmatization**

Lemmatization reduces words to their **base form (lemma)** using **language rules**.

**Example:**

| **Word** | **Lemma** |
| --- | --- |
| running | run |
| better | good |
| mice | mouse |

**Python Code Using NLTK:**

from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

print(lemmatizer.lemmatize("running", pos="v")) # Output: run

**Advantages:** Produces **grammatically correct** words.  
**Disadvantages:** Slower than stemming.

**6. Explain Stemming**

Stemming reduces words to their **root form** by chopping off suffixes without considering meaning.

**Example:**

| **Word** | **Stem** |
| --- | --- |
| running | run |
| studies | studi |
| happiness | happi |

**Python Code Using NLTK:**

from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

print(stemmer.stem("running")) # Output: run

**Advantages:** Faster than lemmatization.  
**Disadvantages:** Sometimes **alters meaning** (e.g., "better" → "bet").

**7. Explain Part-of-Speech (POS) Tagging**

POS tagging assigns **grammatical categories** to words in a sentence.

**Example:**  
Sentence: "The cat sat on the mat."  
POS Tags: [("The", DT), ("cat", NN), ("sat", VBD), ("on", IN), ("the", DT), ("mat", NN)]

**Python Code Using NLTK:**

import nltk

nltk.download("averaged\_perceptron\_tagger")

text = nltk.word\_tokenize("The cat sat on the mat.")

print(nltk.pos\_tag(text))

**Uses:**

* **Named Entity Recognition (NER)**
* **Grammar Checking**

**8. Explain Chunking (Shallow Parsing)**

* Chunking groups **words into meaningful phrases** based on POS tags.
* **Example:**  
  Sentence: "The quick brown fox jumps over the lazy dog."  
  Chunks: [("The quick brown fox", NP), ("jumps", VP), ("over the lazy dog", PP)]
* **Python Code Using NLTK:**

import nltk

sentence = [("The", "DT"), ("big", "JJ"), ("dog", "NN"), ("barked", "VBD")]

grammar = "NP: {<DT>?<JJ>\*<NN>}"

chunk\_parser = nltk.RegexpParser(grammar)

tree = chunk\_parser.parse(sentence)

tree.draw() # Displays the chunk tree

**Uses:** Extracting noun/verb phrases

Preprocessing for Named Entity Recognition (NER)

**9. Explain Noun Phrase (NP) Chunking**

NP chunking **identifies noun phrases** in a sentence.

**Example:**  
Sentence: "The beautiful sunset mesmerized us."  
NP Chunk: [("The beautiful sunset", NP)]

**Usage:**

* Used in **text summarization, question answering, and machine translation**.

grammar = "NP: {<DT>?<JJ>\*<NN>}"

chunk\_parser = nltk.RegexpParser(grammar)

text = [("The", "DT"), ("beautiful", "JJ"), ("sunset", "NN"), ("mesmerized", "VBD")]

tree = chunk\_parser.parse(text)

tree.draw()

**10. Explain Named Entity Recognition (NER)**

NER **identifies and classifies** named entities (persons, organizations, locations, dates) in text.

**Example:**  
Sentence: "Barack Obama was born in Hawaii in 1961."  
NER Output: [("Barack Obama", PERSON), ("Hawaii", LOCATION), ("1961", DATE)]

**Python Code Using SpaCy:**

import spacy

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

text = "Barack Obama was born in Hawaii in 1961."

doc = nlp(text)

for ent in doc.ents:

print(ent.text, ent.label\_)