What is NLP?

- **Definition**: NLP is a field of AI that helps computers **understand, interpret, and generate human language**.
- It is at the intersection of **Linguistics + Computer Science + AI**.
- **Analogy**: NLP is like teaching a machine how to read, listen, and talk like humans.

Why NLP is Important

- Humans speak in **natural languages** (English, Hindi, Marathi...).
- Computers only understand **binary / structured data**.
- NLP acts as a **bridge** between human language and machine language.

Examples in Daily Life

- Google Translate 🌍
- Siri / Alexa 🎙️
- ChatGPT 💬
- Spam filters 📧
- Autocorrect / Predictive text on phones 📱

Challenges in NLP

- **Ambiguity**:
- *"I saw a man on a hill with a telescope."* (Who has the telescope?)
- **Context**:
- *"Book a ticket"* (verb) vs. *"Read a book"* (noun).
- **Sarcasm / Emotion**:
- *"Great! Another Monday!"* (Actually negative).
- **Spelling / Grammar mistakes**.

Components of NLP

- **Text Preprocessing**: Tokenization, Stop word removal.
- **Morphology**: Word structure (teach → teacher).
- **Syntax**: Grammar structure of sentences.
- **Semantics**: Meaning of words/sentences.
- **Pragmatics**: Context + Real-world knowledge.

Why Preprocessing?

- Raw text = **messy** (punctuation, casing, spelling errors).
- Computers can’t process sentences like humans → we must **clean & normalize**.
- Preprocessing = preparing text for analysis.

Main Steps in Preprocessing

Lowercasing

•Convert all words to lowercase.

•"Natural" → "natural"

Removing Punctuation & Special Characters

•"Hello!!!" → "Hello"

Stopword Removal

•Stopwords = very common words with little meaning (is, are, the, in).

•"The cat is running" → "cat running"

Tokenization

•Split text into smaller units (words/sentences).

•"I love NLP" → ["I", "love", "NLP"]

Stemming

•Crude way of reducing words to base/root form.

•"studies → studi", "running → run"

Lemmatization

•Smarter method using dictionary rules.

•"studies → study", "better → good"

Why These Steps Matter

- Reduces complexity of text.
- Ensures different forms of the same word are treated as one.
- Improves performance of NLP tasks like classification, sentiment analysis, search.

Summary

- Preprocessing is the **foundation of NLP**.
- Key steps: Lowercasing → Cleaning → Stopword removal → Tokenization → Stemming/Lemmatization.
- Goal = convert raw messy text into a **structured, analyzable format**.

Stemming vs. Lemmatization

Stemming

- **Definition**: Cutting off prefixes/suffixes to get the root form of a word.
- **Rule-based, mechanical** → often not meaningful.
- **Examples**:
- *"playing"* → *"play"*
- *"studies"* → *"studi"*
- *"happiness"* → *"happi"*
- **Pros**: Fast, simple.
- **Cons**: Output is sometimes not a valid word.

Lemmatization

- **Definition**: Reducing a word to its dictionary base form (**lemma**) using vocabulary + grammar rules.
- **Smarter than stemming**.
- **Examples**:
- *"playing"* → *"play"*
- *"studies"* → *"study"*
- *"better"* → *"good"*
- **Pros**: Always returns valid words.
- **Cons**: Slower (needs dictionary + POS tagging).

Morphology

What is Morphology?

- **Definition**: Study of the **internal structure of words**.
- Words are made of **morphemes** = smallest units of meaning.

Example: *"unhappiness"*

- un- (prefix, "not")
- happy (root, free morpheme)
- -ness (suffix, "state of")

Types of Morphemes

- **Free Morpheme** → can stand alone (*book, play, happy*).
- **Bound Morpheme** → must attach to another (*un-, -ed, -ness*).

Derivational Morphology

- **Purpose**: Creates a **new word** or changes word class.
- **Examples**:
- teach → teacher (verb → noun)
- happy → happiness (adjective → noun)
- kind → unkind (adds new meaning)

Inflectional Morphology

•**Purpose**: Adds **grammatical information**, but does **not** change word class.

•**Examples**:
•play → plays, played, playing
•cat → cats (plural)
•big → bigger, biggest (comparative, superlative)

Phonology: Sound System of Language

- Phonology is the study of **speech sounds** and how they are organized in a language.
- It looks at both **physical sounds** and their **mental representation.**

Key Concepts in Phonology

1. **Phoneme**
   - The smallest unit of sound that can change meaning.
   - Example:
     - /p/ vs /b/ → "pat" vs "bat" (different meaning due to one sound).
2. **Allophones**
   - Variations of the same phoneme that do not change meaning.
   - Example:
     - English /p/ in "pin" (aspirated [pʰ]) vs "spin" (unaspirated [p]) → both are /p/.
3. **Minimal Pairs**
   - Words differing by **only one phoneme**.
   - Example: *sip* vs *zip*, *bat* vs *pat*.
4. **Syllables**
   - Basic units of speech sounds, usually consisting of a **vowel** (nucleus) and optional consonants.
   - Example: "computer" = com-pu-ter (3 syllables).
5. **Prosody (Suprasegmentals)**
   - Features beyond individual sounds:
     - **Stress** (PREsent vs preSENT).
     - **Intonation** (rising tone in questions).
     - **Rhythm** (timing patterns).

Applications of Phonology in NLP

- **Speech Recognition** → mapping sounds to text.
- **Text-to-Speech (TTS)** → generating natural-sounding speech.
- **Accent/Dialect Analysis** → detecting variations in pronunciation.
- **Spelling Correction & Transliteration** → based on sound patterns.

Syntax (Sentence Structure / Grammar)

- Rules for combining words into phrases and sentences.

- Example:
- English: **Subject + Verb + Object** → "The cat chased the mouse."
- Hindi: **Subject + Object + Verb** → "बिल्ली ने चूहा पकड़ा।"

- In NLP → **Parsing, POS tagging, grammar checking**.

Semantics (Meaning of Language)

- Deals with **meaning of words and sentences**.
- Types:
  - **Lexical semantics** → meaning of words (bank = river bank / financial bank).
  - **Compositional semantics** → meaning of phrases/sentences.

- In NLP → **Word sense disambiguation, question answering, machine translation**.

Hierarchy (How they connect)

- Phonology → Morphology → Syntax → Semantics
  (sounds) (words) (sentences) (meaning)

Part-of-Speech (POS) Tagging

What is POS Tagging?

•**Definition**: Assigning each word in a sentence its grammatical role (Noun, Verb, Adjective, etc.).

Example:

- •Sentence: *"Cats run fast."*
- •Tags: Cats = Noun, run = Verb, fast = Adverb.

Main POS Categories

- **Noun (N)** – names a person/place/thing (*cat, book, India*).
- **Verb (V)** – action/state (*run, is, play*).
- **Adjective (ADJ)** – describes a noun (*big, red, smart*).
- **Adverb (ADV)** – describes a verb/adjective (*quickly, very*).
- **Pronoun (PRON)** – replaces a noun (*he, she, they*).
- **Preposition (PREP)** – shows relation (*in, on, with*).
- **Conjunction (CONJ)** – joins words/clauses (*and, but, or*).
- **Determiner (DET)** – specifies noun (*a, the, some*).
- **Interjection (INTJ)** – expressions (*oh!, wow!*)

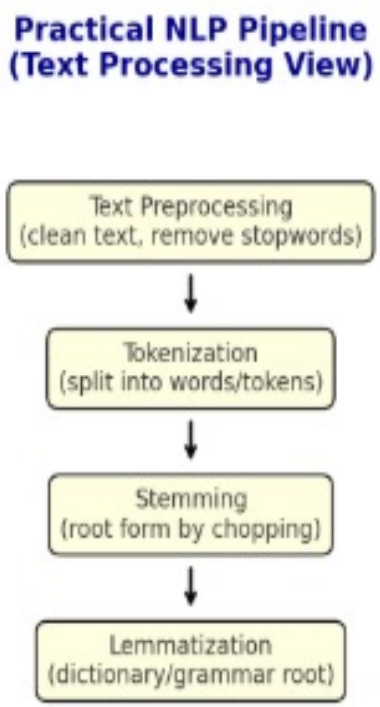
Why POS Tagging is Important

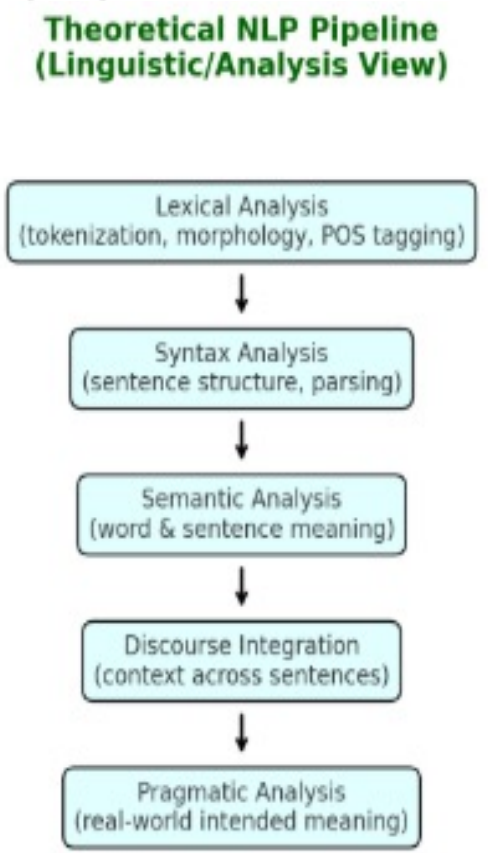
- Helps computers **understand meaning**:
- *"Book a ticket"* (book = verb) vs. *"Read a book"* (book = noun).
- Used in:
- **Parsing** (building sentence structure).
- **Information extraction** (finding names, places).
- **Machine translation**.

Example

- Sentence: *“The little boy is playing in the garden.”*
- The → Determiner
- little → Adjective
- boy → Noun
- is → Verb
- playing → Verb
- in → Preposition
- the → Determiner
- garden → Noun

- **Practical pipeline** = "How we clean text for machines."
- **Theoretical pipeline** = "How machines actually understand language step by step."





Lexical Analysis (Word-Level Analysis)

- Focus: Words
- Breaks text into tokens (words).
- Identifies their base forms (stemming/lemmatization).
- Assigns **Part-of-Speech (POS)** tags (noun, verb, adjective, etc.).
- **Example:**
  Sentence → *"Cats are running fast."*
  Lexical analysis →
- "Cats" → lemma: *cat*, POS: noun (plural)
- "are" → lemma: *be*, POS: verb (auxiliary)
- "running" → lemma: *run*, POS: verb (progressive)
- **Use in NLP:**
  Search engines, spell checkers, word embeddings.

Syntax Analysis (Sentence Structure)

- Syntax Analysis (Sentence Structure)

Syntax analysis is the process of analyzing a sentence to understand its **structure** (how words are arranged and related to each other) according to the rules of grammar.
It checks **“Does this sentence follow the correct grammatical rules?”**
In **Natural Language Processing (NLP)** and **Linguistics**, syntax analysis is also called **parsing**.

In **syntax analysis** (also called *parsing*), we definitely use the concept of a **parse tree**.

🔹 What is a Parse Tree?

A **parse tree** (or derivation tree) is a hierarchical tree structure that shows how a sentence (string of tokens) is generated from the start symbol of a grammar using the production rules of a context-free grammar (CFG).

It represents:

- **Non-terminals** → internal nodes (e.g., *Expr, Statement*)
- **Terminals (tokens)** → leaf nodes (e.g., `id`, `+`, `*`, `num`)
- **Root** → start symbol of grammar

Grammar:

```
r

E → E + T | T
T → T * F | F
F → (E) | id
```

Input string:

```
bash

id + id * id
```

Parse Tree (simplified structure):

```
r

          E
         /|\
        E + T
        |   /|\
        T  T * F
        |  |   |
        F  F   id
        |  |
       id id
```

Difference between Parse tree and Abstract parse tree

- Also called **Concrete Syntax Tree**.
- Directly follows the grammar production rules.
- Contains **all non-terminals and terminals**.
- Shows **every detail** of derivation (even redundant nodes).
- **Bigger, more detailed, but sometimes cluttered.**

✅ **Use:** To check if the sentence is grammatically correct.

Example: For expression `id + id * id`

r

Copy coc

```
        E
       /|\
      E + T
      |   /|\
      T  T * F
      |  |   |
      F  F   id
      |  |
     id id
```

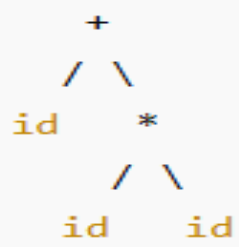
Abstract Parse Tree

- Simplified form of the parse tree.
- Removes unnecessary grammar symbols (like `E`, `T`, `F`).
- Only keeps **essential structure** of the expression/program.
- Internal nodes → **operators or constructs**
- Leaf nodes → **operands (identifiers, constants, literals)**

✅ **Use:** For semantic analysis, optimization, and code generation.

Example: For expression `id + id * id`

bash



✨ Key Differences (Quick Table)

Feature	Parse Tree	Syntax Tree (AST)
Shows	Complete derivation	Essential structure
Size	Larger, detailed	Smaller, compact
Contains	Terminals + Non-terminals	Only operators + operands
Use	Syntax checking	Semantic analysis, code generation

What is a Context-Free Grammar (CFG)?

- A **CFG** is a set of production rules that describe how sentences in a language are formed.

- Widely used in **Compiler Design** (to describe programming languages) and in **NLP** (to describe natural language syntax).

A CFG is defined as a 4-tuple:

$$G = (V, \Sigma, R, S)$$

where:

- **V** → set of **non-terminal symbols** (like `S`, `NP`, `VP`, etc.)
- **Σ** → set of **terminal symbols** (actual words like *cat*, *sat*, *on*)
- **R** → set of **production rules** (like `S → NP VP`)
- **S** → start symbol (usually `S`, the sentence)

Breakdown of Your Grammar

nginx

```
S  → NP VP       # A sentence is made of a Noun Phrase + Verb Phrase
NP → Det N       # A noun phrase is a Determiner + Noun
VP → V PP | V    # A verb phrase can be: Verb + Prepositional Phrase OR just Verb
PP → P NP        # A prepositional phrase is a Preposition + Noun Phrase
Det → "the"      # Determiner is the word "the"
N → "cat" | "mat"  # Noun can be "cat" or "mat"
V → "sat"        # Verb is "sat"
P → "on"         # Preposition is "on"
```

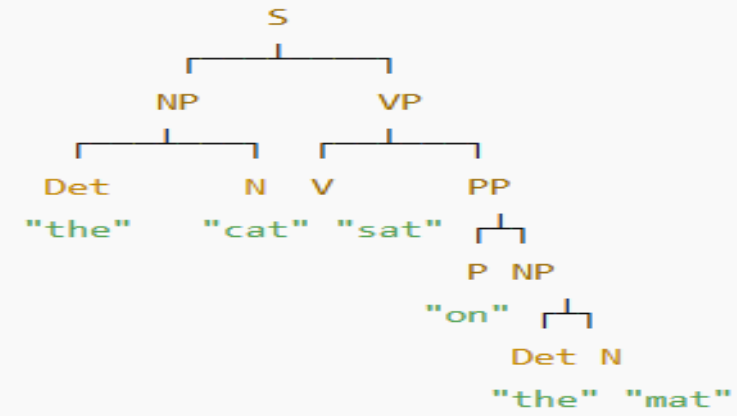
Parse Tree using CFG

Sentence: **"the cat sat on the mat"**

1. Start with `S`
2. Apply `S → NP VP`
3. Expand `NP → Det N` → ("the cat")
4. Expand `VP → V PP` → ("sat on the mat")
5. Expand `PP → P NP`

👉 Parse tree structure:

mathematica



Semantic Analysis (Meaning of Sentences)

- **Definition:**
  Semantic analysis is the process of **extracting meaning** from sentences.
  It focuses on the **"what does this sentence actually mean?"** part, beyond just grammar.
- While **syntax** checks if the structure is correct, **semantics** ensures that the sentence **makes sense logically and meaningfully**.

Goals of Semantic Analysis

- Understand the **meaning of words** in context (lexical semantics).
- Understand how **word meanings combine** to form **sentence meaning** (compositional semantics).
- Resolve **ambiguities** (when words/sentences have multiple meanings).
- Identify **relationships** (who did what, when, where, to whom).

Types of Semantic Analysis

1. Lexical Semantics

- Deals with the **meaning of individual words.**
- Example:
  - *"Bank"* → could mean **river bank** or **financial bank.**
  - Semantic analysis uses context to choose the correct meaning.

3. Word Sense Disambiguation (WSD)

- Choosing the **correct sense of a word** when multiple meanings exist.
- Example: *"I went to the bank to deposit money."* → financial institution
- *"I sat on the bank of the river."* → river side

---

4. Semantic Role Labeling (SRL)

- Finds **roles** in a sentence: who is the **doer (agent)**, what is the **action**, and what is the **receiver (object/patient)**.
- Example:
  - Sentence: *"Ram gave a book to Sita."*
  - Agent: Ram (giver)
  - Theme: book (thing given)
  - Recipient: Sita (receiver)

---

5. Handling Ambiguity

Semantic analysis resolves **ambiguities**:

- *"Visiting relatives can be annoying."*
  - Meaning 1: It is annoying to visit relatives.
  - Meaning 2: Relatives who are visiting can be annoying.

Applications of Semantic Analysis

- **Machine Translation** (Google Translate)
- **Chatbots / Virtual Assistants**
- **Search Engines** (understanding query meaning)
- **Text Summarization**
- **Question Answering Systems**
- **Sentiment Analysis** (positive/negative/neutral meaning)

Discourse Integration (Context Across Sentences)

- Discourse Integration is the process of understanding **how the meaning of one sentence depends on, connects to, and contributes to the meaning of other sentences** in a text or conversation.
- It ensures that sentences are not interpreted **in isolation**, but rather as part of a **larger context (discourse)**.

Why It’s Needed?

- In real life, meaning is not just in single sentences.
- Many sentences rely on **previous context** to make sense.
- Without discourse, AI/chatbots would misunderstand conversations.

Key Features of Discourse Integration

1. **Anaphora Resolution (Pronoun Resolution)**
   - Figuring out what pronouns like *he, she, it, they, this, that* refer to.
   - Example:
     - Sentence 1: *"Sita bought a book."*
     - Sentence 2: *"She read it quickly."*
     - **Discourse integration:** *She → Sita, It → book.*

---

2. **Ellipsis Resolution (Filling in Missing Words)**
   - Sometimes words are left out because context makes them clear.
   - Example:
     - A: *"Do you want tea?"*
     - B: *"Yes, I do."* (*I do = I want tea.*)

---

3. **Coherence and Cohesion**
   - Ensuring sentences are logically connected.
   - Example:
     - *"It was raining. We stayed indoors."* → Coherent.
     - *"It was raining. The sun is very hot."* → Not coherent.

4. **Discourse Relations**
    - Sentences are connected with relations like:
        - Cause-effect: *"He fell because it was slippery."*
        - Contrast: *"It rained, but we played cricket."*
        - Temporal (time): *"She cooked dinner, then we ate."*

---

5. **Presupposition & Implicature**
    - Understanding **implied meaning** beyond explicit words.
    - Example:
        - *"John stopped smoking."* → Presupposes that John **used to smoke**.
        - *"Can you pass the salt?"* → Implicature = Request, not a question about ability.

Pragmatic Analysis (Real-World Context)

- Pragmatic Analysis is the process of interpreting sentences in relation to the **real-world context, speaker's intention, and situation** in which they are used.

Why Needed?

- A sentence may have multiple meanings depending on the **context**.
- Syntax + semantics may be correct, but without pragmatics, we cannot capture the **intended meaning**.

Key Aspects of Pragmatic Analysis

1. Speaker's Intention (Illocutionary Meaning)

- Example: *"Can you open the window?"*
  - Semantics: A question about ability.
  - Pragmatics: A polite **request** to open the window.

---

2. Deixis (Context-Dependent Words)

- Words whose meaning depends on the context of utterance: *here, there, I, you, this, that, tomorrow.*
- Example: *"I will meet you here tomorrow."*
  - "I" → depends on speaker.
  - "you" → depends on listener.
  - "here" → depends on location.
  - "tomorrow" → depends on date of utterance.

---

3. Conversational Implicature

- Meaning implied but not directly stated.
- Example:
  - A: *"Are you coming to the party?"*
  - B: *"I have to work tomorrow."*
  - Semantics: Statement about work.
  - Pragmatics: Implies **B will not attend the pa**.

- **Sentence** → a grammatically complete structure (written with rules).
- **Utterance** → what someone actually says/writes in real communication (may or may not be grammatically correct).

Example:

- Sentence: *"I am going to school."*
- Utterance: *"Goin' school now."* (not perfectly grammatical, but still meaningful in context).

4. Speech Acts

- Utterances can perform **actions**:
  - Assertion: *"It's raining."*
  - Request: *"Please pass the salt."*
  - Command: *"Close the door."*
  - Promise: *"I'll call you tomorrow."*
  - Question: *"What time is it?"*

---

5. Politeness and Social Context

- Pragmatics considers **tone, culture, and social rules**.
- Example: Saying *"Could you please..."* instead of *"Do this."* is politeness strategy.

Summary

Step	Focus	Example	NLP Use
Lexical Analysis	Words	“running” → run (verb)	Tokenization, POS tagging
Syntax Analysis	Grammar	Parse tree of “The cat chased the mouse”	Grammar checking, parsing
Semantic Analysis	Meaning	Bank = riverbank (not financial)	Machine translation, QA
Discourse Integration	Context	“Ravi bought a car. It is red.” → “It” = car	Summarization, dialogue
Pragmatic Analysis	Real-world intent	“Can you open the window?” = request	Chatbots, sarcasm detection