SAPTHAGIRI NPS UNIVERSITY

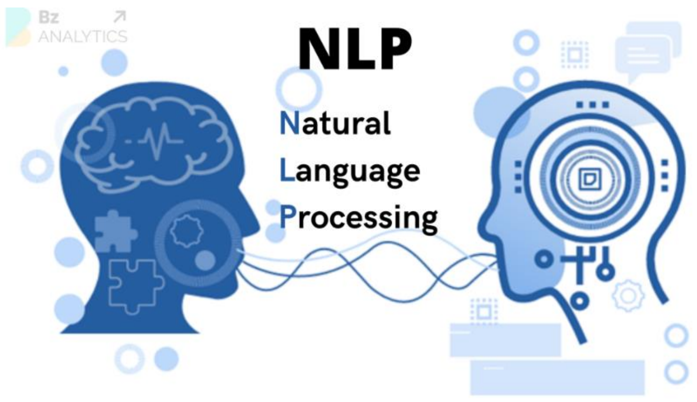
UNMATCHED EXCELLENCE, UNLIMITED POTENTIAL

1st Semester

Fundamentals of AI and ML (24BTPHY106)

Module-5
Natural Language Processing

Language Models - Grammar - Parsing - Augmented Grammars - Complications of Real Natural Language - Natural Language Tasks



Natural Language Processing

Natural Language Processing (NLP) is a branch of Artificial Intelligence (AI) that focuses on enabling computers to understand, interpret, and respond to human language in a meaningful way. It combines linguistics and computer science to bridge the gap between human communication and machine understanding.

It enables machines to:

- Understand: Extract meaning from human language.
- Process: Analyse and manipulate text or speech.
- Generate: Produce language outputs (e.g., text or speech).

It aims to make human-computer interaction seamless by bridging the gap between human communication (natural language) and computer language (binary or structured data).

Natural Language Processing (NLP) is essential because it enables the analysis and understanding of vast amounts of unstructured data. In today's digital world, most of the data generated—such as text from emails, social media posts, customer reviews—is unstructured, making it challenging for traditional systems to process.

With NLP, businesses and organizations can analyse this data to extract meaningful insights, extract meaning from human language such as identifying trends, sentiments, or key topics.

Moreover, NLP powers automation in various domains, including chatbots, virtual assistants, and voice-to-text systems, allowing for seamless interaction between humans and machines.

For example, when you ask Siri or Alexa a question, NLP helps them understand what you are saying and figure out the best answer to give you. This makes it possible for you to have a smooth and natural conversation with these AI assistants.

Applications of NLP

- **Enhancing Human-Computer Interaction**
  Natural Language Processing (NLP) enables efficient communication between humans and computers, making interactions feel natural and user-friendly.
- **Gaining Insights from Unstructured Data**
  NLP helps analyse vast amounts of unstructured text data, such as social media posts, reviews, and emails, extracting valuable insights and trends that inform decision-making.
- **Automating Language-Based Tasks**
  NLP can do jobs that involve understanding language, like figuring out if a message is positive or negative, making a short summary of a long article. This saves time because computers do these tasks faster than people.
- **Intelligent tutoring systems**
  They help students learn languages by analyzing their writing or speaking. These systems give feedback on grammar, vocabulary, and pronunciation. They can also create conversations for students to practice speaking in real-life situations.
- **Automated essay grading systems**
  It helps teachers by checking students' essays. They look at grammar, sentence flow, and the strength of ideas. These systems give quick feedback

so students can improve their writing. This makes learning faster, more personalized, and helpful for both students and teachers.

- **Clinical documentation**
  Hospitals use NLP to create reports automatically from patient data, saving doctors time and reducing paperwork. For example, it can turn notes from a doctor's visit into a detailed medical report.
- In **medical research**, NLP helps by analyzing large amounts of research papers to find useful information. This process, called text mining, allows researchers to quickly discover patterns, trends, and insights from scientific studies.

Two use cases of NLP

i. Sentiment Analysis

Sentiment Analysis Sentiment analysis, also known as opinion mining, is an NLP application that involves determining the sentiment or emotional tone behind a body of text. It is commonly used in social media monitoring, customer feedback analysis, and product reviews to gauge public opinion or customer satisfaction.

Use Case: Imagine a company launching a new product. By analysing social media posts, reviews, and comments, sentiment analysis can determine whether the general public's response is positive, negative, or neutral. This helps the company to understand customer satisfaction, identify potential issues, and even tailor marketing strategies accordingly.

How it Works: The process involves breaking down the text into smaller parts (tokenization), removing irrelevant words (stop words), and then analysing the remaining text to detect sentiments (positive, negative, neutral). Machine learning models are often trained on labelled datasets to predict sentiments accurately.

ii. Chatbots and Virtual Assistants

Chatbots and virtual assistants like Siri, Alexa, and Google Assistant use NLP to interact with users in natural language.

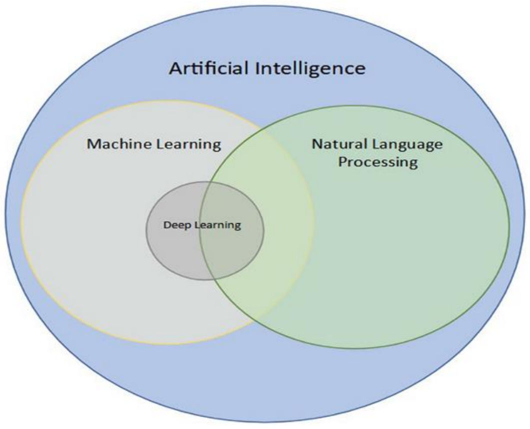
Use Case: Imagine a customer service chatbot that helps people fix their devices. When a user explains their problem in everyday language, the chatbot understands and gives clear, step-by-step instructions. This means customers get faster help without needing to wait for a human assistant, making their experience better.

How it Works: NLP makes this possible by:

NLP uses techniques such as entity recognition **(**Finding important details, like names or dates, in the user's message). Intent recognition (Understanding what the user wants to do), Dialogue management (Keeping the conversation smooth and organized).

These techniques help chatbots respond like humans and make communication easy.

How are AI, ML and NLP related?



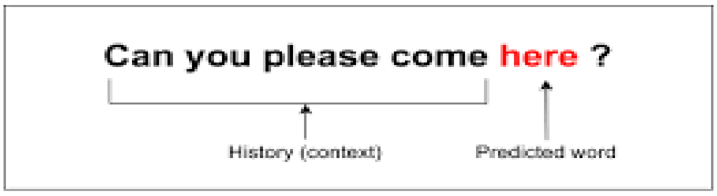
Artificial Intelligence is all about creating systems that can do things that normally need human intelligence, like solving problems, understanding speech, or recognizing faces in photos.

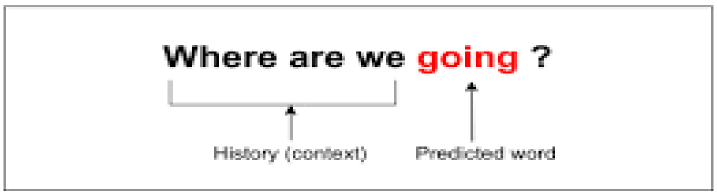
Machine Learning is a part of AI. Think of it to teach computers to learn from experience. Instead of programming every single step, you give the computer lots of examples, and it figures out the patterns and learns how to make decisions or predictions on its own.

Natural Learning Processing is a part of both AI and ML. It focuses on teaching computers to understand and use human language.

1. LANGUAGE MODELS

A language model is a type of machine learning model that guesses the next word in a sentence based on the words that came before it. In simple terms, it looks at the context of a sentence and predicts which word fits best in a blank space.





The important thing is that the model does not focus on grammar, but rather on how words are used in a way that is like how people write.

How does Language Model Work?

Language models (LMs) are systems designed to understand and generate human-like text based on the input they receive.

The process of how the language model works is given below:

i. Training Phase

Language models are trained on large datasets of text to learn patterns in language. The training process includes the following steps:

a. Tokenization: Text is split into smaller units, called tokens, which can be words, subwords, or characters. For example, "Language Model" might be tokenized as ["Language", "Model"] or ["Lang", "uage", "Model"].

b. Learning Probabilities: The model learns the probability of a token given the previous tokens. For example:

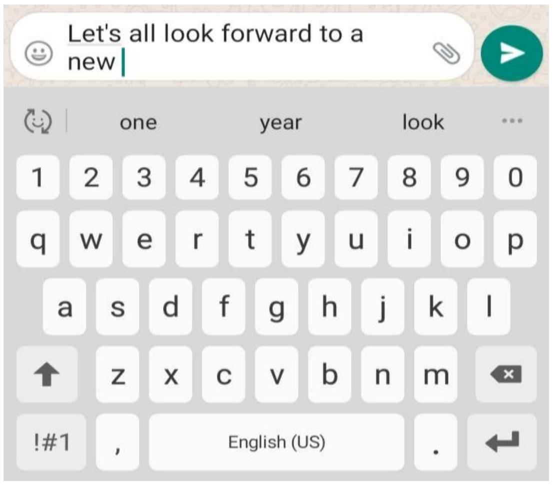
- In "The cat sat on the ____," the model learns that "mat" is more likely to follow than "sky."

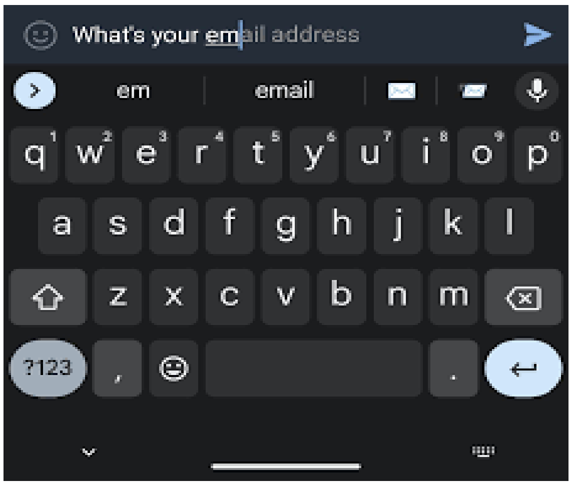
c. Neural Networks: Modern LMs, like GPT (Generative Pre-trained Transformer), They process data in layers, adjusting millions (or billions) of parameters to minimize prediction errors.

ii. Generation Phase

When generating text prompt or starting text is provided as input. The model predicts the most probable next token repeatedly to form meaningful sentences.

Have you ever noticed the smart features in Google Gboard and Microsoft SwiftKey keyboards that provide auto-suggestions to complete sentences when writing text messages?





Applications of Language Model

- **Chatbots and Virtual Assistants**

Virtual Assistants like Siri, Alexa, and Google Assistant use Language models to predict the next-word to interpret voice commands and generate appropriate responses.

Example: If a user says, "What’s the weather like today?" the assistant might predict the next words like "in New York" based on previous interactions or context.

- **Autocompletion and Autocorrection**

Text editors, search engines, and messaging apps use language models to suggest the next word as you type. This helps users complete sentences faster or correct typos.

- **Content Generation**

LMs are used for creating content, such as writing articles, blog posts, or even poetry. They generate human-like text based on a given topic.

- **Speech Recognition**

LMs help convert spoken language into written text by understanding the context and predicting what the speaker is saying, improving accuracy.

- **Personalized Recommendations**

LMs analyze text data (like browsing history or past interactions) to recommend personalized products, movies, or content.

- **Code Generation and Assistance**

LMs can assist in programming by suggesting code snippets, detecting errors, and even generating code based on natural language descriptions (e.g., GitHub Copilot).

- **Social Media Content Creation**

Influencers and marketers use LMs to generate catchy headlines, posts, or hashtags to engage audiences on social media platforms like Twitter or Instagram.

- **Grammar and Style Checking**

Tools like Grammarly use LMs to check for spelling, grammar, punctuation, and style issues in written text, helping users improve their writing.

1.1. The bag-of-words model

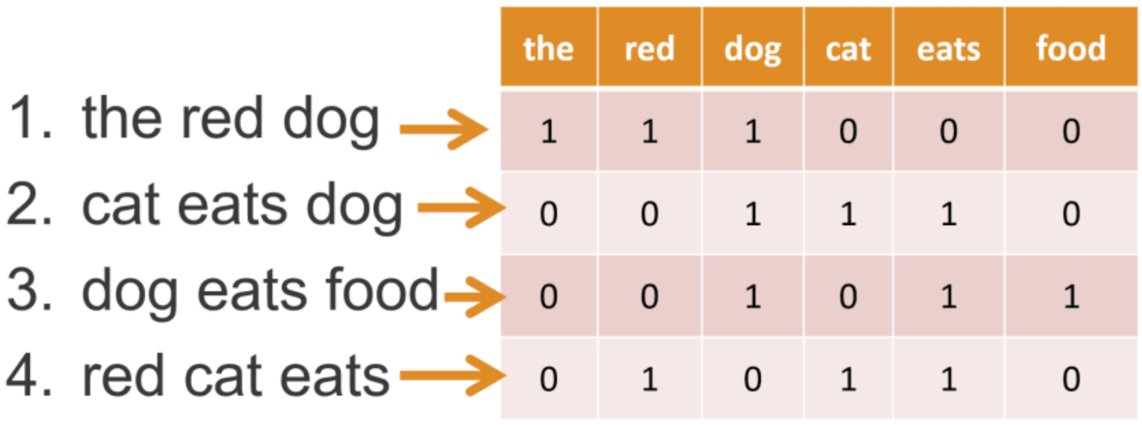
Whenever we apply any algorithm in NLP, it works on numbers. We cannot directly feed our text into that algorithm. Hence The Bag of Words (BoW) model is a way to transform text into a set of numbers. It does this by creating a list (or "bag") of all the words in the text and counting how many times each word appears.

This model can be visualized using a table, which contains the count of words corresponding to the word itself.

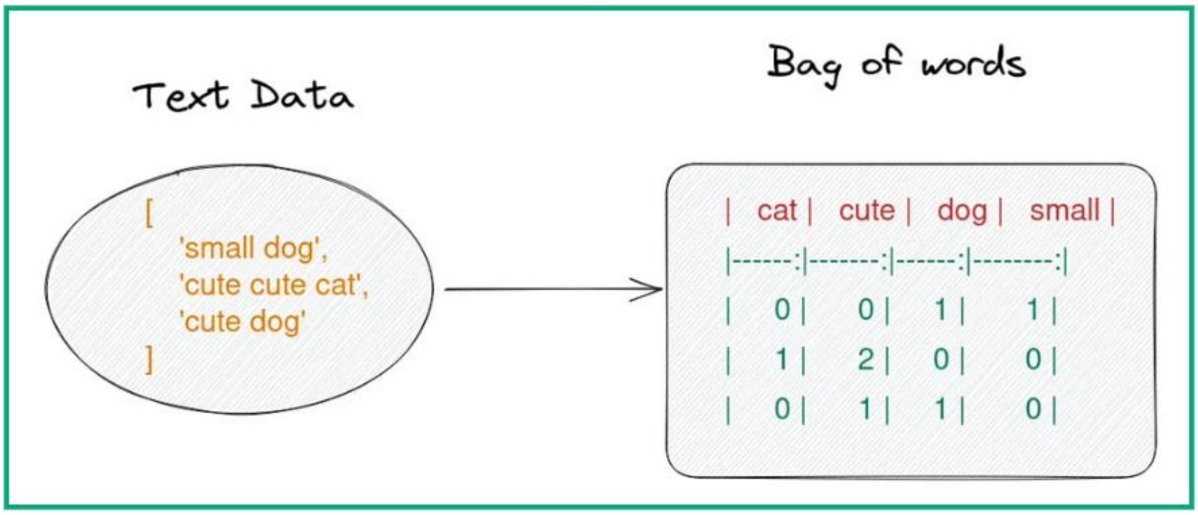
It involves two things:

- A vocabulary of known words.
- A measure of the presence of known words.

Example:



	the	red	dog	cat	eats	food
1. the red dog	1	1	1	0	0	0
2. cat eats dog	0	0	1	1	1	0
3. dog eats food	0	0	1	0	1	1
4. red cat eats	0	1	0	1	1	0



```
[
  'small dog',
  'cute cute cat',
  'cute dog'
]
```

```
| cat | cute | dog | small |
|------:|-------:|------:|--------:|
|    0|     0|    1|      1|
|    1|     2|    0|      0|
|    0|     1|    1|      0|
```

<u>**Key Features of the Bag of Words Model:**</u>

a. **Tokenization**: The first step is to break down the text into individual words (tokens).
Example: "I love programming" → ["I", "love", "programming"]

b. **Vocabulary**: A set of all unique words (vocabulary) across the entire dataset is created. Each word is assigned an index.

Example: Vocabulary: ["I", "love", "programming"]

c. **Word Frequency Count**: For each document (or sentence), count how many times each word in the vocabulary appears.

Example: For the document "I love programming":

- "I" appears 1 time.
- "love" appears 1 time.
- "programming" appears 1 time.

d. **Vector Representation**: Each document is represented as a vector with the word counts (or binary values indicating the presence of a word).

   Example: If the vocabulary is ["I", "love", "programming"], the sentence "I love programming" is represented as:

   - [1, 1, 1] (since each word appears once).

The disadvantages of Bag-of-word model is that there is no Contextual Information:

- BoW does not consider the order or context of words. For example, "The cat sat on the mat" and "On the mat sat the cat" would be represented the same, despite having the same words in a different order.

High Dimensionality:

The vocabulary can grow very large, especially with diverse text. This leads to high-dimensional vectors, which may result in memory and computation inefficiencies.

Example: If the dataset contains many rare words, the vector representation becomes sparse, meaning many dimensions are unused.

1.2. n-gram model

The n-gram model is a simple yet powerful tool used in Natural Language Processing (NLP) to understand text. Instead of looking at single words, like the Bag of Words model, the n gram model looks at sequences of words to capture more context.

A good N-gram model can predict the next word in the sentence. There are different n-gram models:

Unigram (1-gram)

A unigram model looks at each word independently, treating them as individual units without considering any context.

Given the sentence "I love programming," the unigram model would focus on the individual words: "I," "love," "programming."

The probability of each word would be calculated based on its frequency.

Bigram (2-gram)

A bigram model considers pairs of consecutive words. It models the probability of a word occurring based on the word that immediately precedes it. It captures some context, as it looks at the relationship between two adjacent words.

Example:

- Sentence: "I love programming."
- Bigrams: ("I love"), ("love programming")
- The probability of "love" given "I" and the probability of "programming" given "love" would be calculated.

Trigram (3-gram)

A trigram model considers three consecutive words. It calculates the probability of the third word based on the previous two words. This gives the model a better understanding of the context by looking at a larger window of words.

Sentence: "I love programming a lot."

Trigrams: ("I love programming"), ("love programming a"), ("programming a lot")

The probability of "programming" given "I love", "a" given "programming", and "lot" given "a" and "programming" would be calculated.

Higher-order N-grams (4-gram, 5-gram, etc.)

Higher-order N-grams (e.g., 4-grams, 5-grams) extend the idea of bigrams and trigrams by considering even longer sequences of words. A 4-gram considers four consecutive words, a 5-gram considers five, and so on.

This is Big Data AI Book

Uni-Gram	This	Is	Big	Data	AI	Book
Bi-Gram	This is	Is Big	Big Data	Data AI	AI Book	
Tri-Gram	This is Big	Is Big Data	Big Data AI	Data AI Book		

1.3. Applications of n-gram model

Text Generation

N-gram models can be used to generate text by predicting the next word in a sequence based on the preceding *n-1* words. Starting from an initial word or phrase, the model generates the next word, then uses that word to generate the following one, and so on.

Speech Recognition

In speech recognition, N-gram models help predict the next word based on the context of the previous words. For example, when converting spoken language into text, the model can use N-grams to help resolve ambiguity in speech (e.g., homophones like "see" and "sea").

Plagiarism Detection

N-gram models are used in plagiarism detection systems to compare text and identify overlapping word sequences. The presence of similar N-grams across different documents could suggest plagiarism or content duplication.

Text Classification: Spam detection, sentiment analysis, topic categorization

Autocompletion and Prediction: Predictive typing, code autocompletion, text suggestions.

Spell Checking and Correction: Word processors, search engines, social media platforms.

Information Retrieval: Web search, document retrieval based on query understanding.

1.4. Smoothing n-gram models

When we use **n-gram models** to predict the next word in a sentence, we face a common issue:

- What if a combination of words (n-gram) in the test data was never seen in the training data?
- The model will assign a zero probability to that n-gram, which is not ideal.

Smoothing helps by giving small probabilities to these unseen n-grams, ensuring that every possible combination has at least some chance of occurring.

Add-One Smoothing (Laplace Smoothing)

Imagine you have a small pizza and someone wants to share it with a new guest. You cut the pizza into smaller slices to include the guest.

Similarly, in Laplace smoothing, we add **1** to the count of every possible n-gram, even those that were not seen before.

- Example:
  If "cat eats" appears 3 times and "cat jumps" appears 0 times, we adjust the counts to:
  - "cat eats": 3 + 1 = 4
  - "cat jumps": 0+1=10 + 1 = 10+1=1

Smoothing ensures that the model does not assign zero probability to unseen word combinations. Makes better predictions, even with limited data.

1.5. Word Representation

In natural language processing (NLP), word representation refers to the way words are encoded into a format that computers can process and understand. Since computers cannot directly interpret text, words need to be converted into numerical or vectorized formats.

Humans understand words based on their meaning, relationships, and context. For example:

- "dog" and "cat" are similar because they are animals.
- "dog" and "bark" are related because dogs bark.

A good word representation captures these relationships and encodes them into a numerical format.

i. **Words as Points in Space**
Imagine each word as a point in a multi-dimensional space:

- Similar words (e.g., "king" and "queen") are closer together.
- Dissimilar words (e.g., "king" and "car") are farther apart.

Example:
In a 3D space, the vectors might look like this:

- "king" → [0.8, 0.2, 0.1]
- "queen" → [0.9, 0.3, 0.2]
- "car" → [0.1, 0.7, 0.8]

Here, "king" and "queen" are closer than "king" and "car."

ii. **Synonyms and Similar Words**
Words with similar meanings have similar vectors.
Example:

"happy" and "joyful" will have similar representations, like:

- "happy" → [0.5, 0.7, 0.1]
- "joyful" → [0.6, 0.6, 0.2]

This is useful for tasks like synonym detection or clustering words with similar meanings.

iii. **Context and Ambiguity**
Word representations can capture how the meaning of a word changes in different contexts.
Example:

"bank" in "river bank" vs. "money bank."

- In "river bank," the representation of "bank" is closer to "river" and "water."
- In "money bank," it is closer to "money" and "finance."

iv. **Categorical Clustering**

Words related to a category naturally cluster together.

Example:

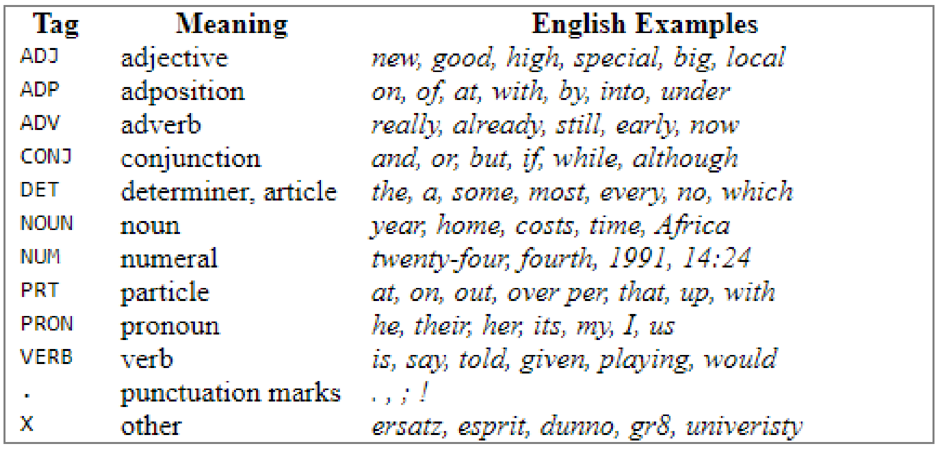
In vector space:

Animals: "cat," "dog," "lion," "tiger" → form a cluster.

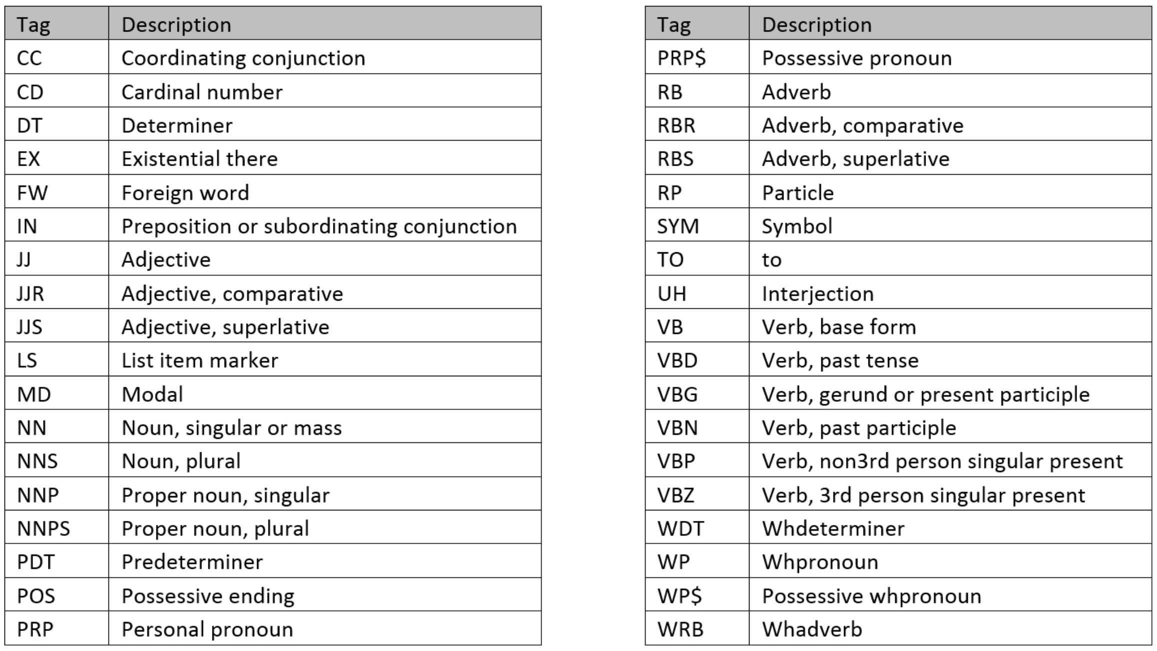
Food: "pizza," "pasta," "burger" → form another cluster.

1.6. Part-of-speech (POS) tagging

Part-of-Speech (POS) tagging is the process of assigning a part-of-speech label (e.g., noun, verb, adjective) to each word in a sentence, based on its definition and context. POS tagging is a fundamental step in natural language processing (NLP) as it helps in understanding the grammatical structure and meaning of a text.



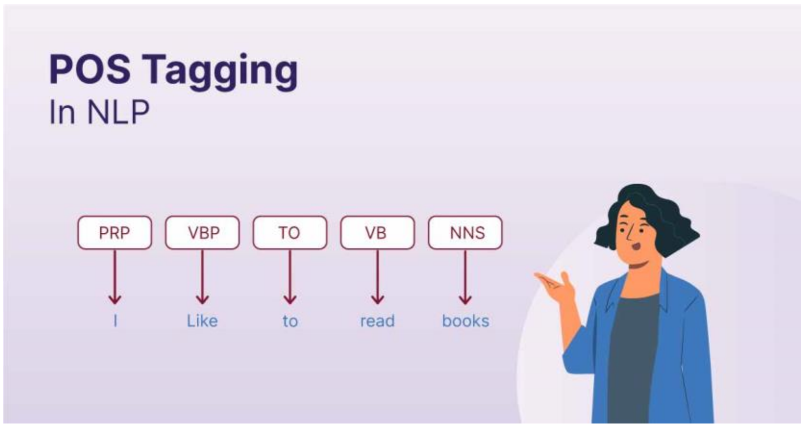
Tag	Meaning	English Examples
ADJ	adjective	*new, good, high, special, big, local*
ADP	adposition	*on, of, at, with, by, into, under*
ADV	adverb	*really, already, still, early, now*
CONJ	conjunction	*and, or, but, if, while, although*
DET	determiner, article	*the, a, some, most, every, no, which*
NOUN	noun	*year, home, costs, time, Africa*
NUM	numeral	*twenty-four, fourth, 1991, 14:24*
PRT	particle	*at, on, out, over per, that, up, with*
PRON	pronoun	*he, their, her, its, my, I, us*
VERB	verb	*is, say, told, given, playing, would*
.	punctuation marks	*. , ; !*
X	other	*ersatz, esprit, dunno, gr8, univeristy*



Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun

Tag	Description
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Whdeterminer
WP	Whpronoun
WP$	Possessive whpronoun
WRB	Whadverb

Figure: Some examples of POS-tags



Part-of-Speech (POS) tagging is the process of identifying and labeling each word in a sentence with its grammatical category, such as noun, verb, adjective, or adverb. It helps computers understand the structure and meaning of sentences. For example, in the sentence **"The dog barks loudly,"** each word has a specific role:

- **"The"** is a determiner (DT).
- **"dog"** is a noun (NN).
- **"barks"** is a verb (VBZ).
- **"loudly"** is an adverb (RB).

By assigning these tags, POS tagging provides insights into how words interact with each other.

This tagging is particularly useful when words have multiple meanings. For instance, consider the word **"book":**

- In **"Book a ticket,"** "book" is a verb because it refers to the action of reserving something.
- In **"This is a book,"** "book" is a noun because it refers to an object.

To decide the correct tag, POS tagging considers the context of the sentence.

POS tagging is essential in many natural language processing (NLP) tasks, such as chatbots, grammar correction tools, and text analysis. For instance, a grammar-checking tool can highlight errors like using a noun where a verb is needed, improving the clarity of writing.

Challenges in POS Tagging

Some common challenges in part-of-speech (POS) tagging include:

Ambiguity: Many words can belong to multiple parts of speech depending on their usage in a sentence. The context determines their correct tag, making it challenging for the model to decide.

Example:

"Book a room." ("book" is a verb)

"This is a book." ("book" is a noun)

Out-of-Vocabulary words: When the model encounters a word it has not seen before (e.g., slang, technical terms, or newly coined words), it struggles to assign the correct POS tag.

Example:

The sentence: "The zoogle was found in the forest."

The word "zoogle" is unknown, making it hard to tag.

Complex Sentence Structures: In long or complex sentences, understanding the relationships between words becomes difficult. Words may depend on other distant words for their correct tags.

Example:

"The boy who was reading the book fell asleep."

The word "reading" depends on "boy" to be identified as a verb.

Differences in Language Variations: Dialects, regional variations, and informal language (e.g., social media text or chat messages) can differ significantly from formal text, making tagging harder.

Example:

Formal: "I am not sure."

Informal: "Idk." ("idk" means "I don't know," but it doesn't fit traditional POS categories.)

Errors in Tokenization: If the text is not properly split into individual words (tokens), it leads to incorrect tagging.

Example:

Input: "John's running late."

Incorrect tokenization may treat "John's" as one word, leading to errors.

1.7. Comparing language models with their characteristics

Unigram (1-gram) Model: Simple and fast, but it's the least effective because it ignores context and treats each word independently.

Bigram (2-gram) Model: Better than unigram because it captures relationships between consecutive words, improving fluency.

Trigram (3-gram) Model: More effective than bigram as it considers the previous two words, providing a richer context for prediction.

4-gram Model: The best for context as it captures longer word dependencies, but it requires more data and computational resources.

2. GRAMMAR

In **Natural Language Processing (NLP)**, **grammar** refers to the set of rules that help machines understand and generate human language. Unlike traditional grammar, which deals with rules for constructing sentences in human languages, NLP grammar focuses on enabling machines to parse, interpret, and produce meaningful text. Here's how grammar plays a key role in NLP:

Sentence: "She quickly ran to the store."

- POS Tags:
  - She (Pronoun)
  - quickly (Adverb)
  - ran (Verb)
  - to (Preposition)
  - the (Determiner)
  - store (Noun)

Here, each word is labeled with its grammatical category (part of speech), allowing the NLP system to understand the structure of the sentence.

Input Sentence: "He go to school every day."

- **NLP Correction:**
  - Corrected Sentence: "He goes to school every day."

In this example, a grammar-checking NLP tool detects that "go" should be "goes" to match the subject "He."

Context-Free Grammar (CFG) is a set of rules used to describe how sentences in a language are structured. It is called "context-free" because the rules apply regardless of the surrounding words or phrases. CFG is widely used in NLP to help computers understand and process language.

Basic Parts of CFG

i. Non-terminal Symbols:
   These are placeholders for parts of a sentence, like *Noun Phrase (NP)* or

Verb Phrase (VP). They represent groups of words or grammatical categories. Example: *S* (Sentence), *NP* (Noun Phrase).

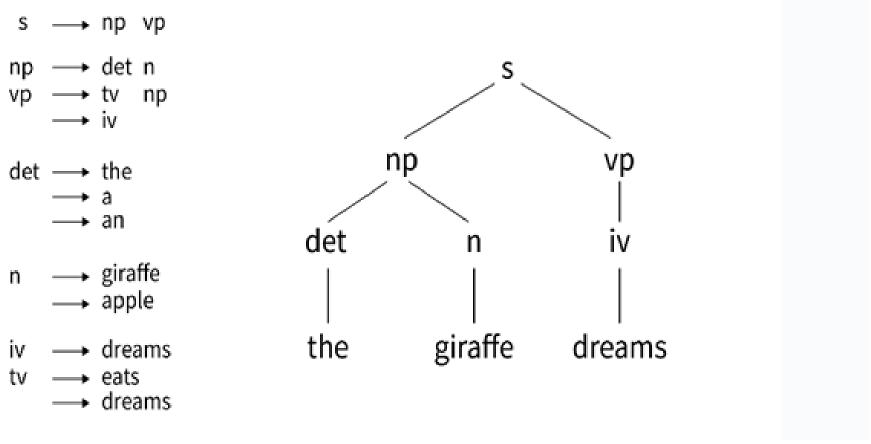
ii. **Terminal Symbols**:
These are the actual words or tokens in a sentence. They can't be broken down further. Example: *dog*, *runs*, *the*.

iii. **Production Rules**:
These are instructions that explain how to form sentences.

Example: *S → NP VP* (A sentence consists of a noun phrase followed by a verb phrase).

iv. **Start Symbol**: The starting point of the grammar, usually *S* for Sentence.

Example: Construct a parse tree for the following sentence using CFG rules: "The giraffe dreams.



```
s    ⟶ np  vp

np   ⟶ det  n
vp   ⟶ tv   np
     ⟶ iv

det  ⟶ the
     ⟶ a
     ⟶ an

n    ⟶ giraffe
     ⟶ apple

iv   ⟶ dreams
tv   ⟶ eats
     ⟶ dreams
```

2.1. The lexicon of ε₀

The **lexicon of ε₀** refers to the list of all possible words or symbols that a system or model can understand and use.

Think of it like a dictionary that defines all the words a language can have. In grammar or computational models, this lexicon includes the basic building blocks (like words or tokens) used to form sentences.

For sentences like *"The boy eats an apple"* or *"A girl plays the piano"*, the lexicon could look like this:

- Articles (Det): *the, a, an*
- Nouns (N): *boy, girl, apple, piano*

- Verbs (V): *eats, plays*
- Prepositions (P): *on, with*

Words are categorized into two main groups based on how they evolve and are used:

Open Classes: Include nouns, names (proper nouns), verbs, adjectives, and adverbs. Words in open classes provide essential content and meaning in sentences.

Closed Classes: Include pronouns, relative pronouns, articles, prepositions, and conjunctions. Words in closed classes serve functional roles in grammar, establishing relationships and structure within sentences.

Uses of Lexicon in NLP:

- A lexicon provides information about the categories of words, such as nouns, verbs, adjectives, etc.
- The lexicon provides a mapping of words to their possible grammatical roles (e.g., noun, verb).

Example: For the sentence *"I run daily"*, the lexicon helps tag *"run"* as a verb.

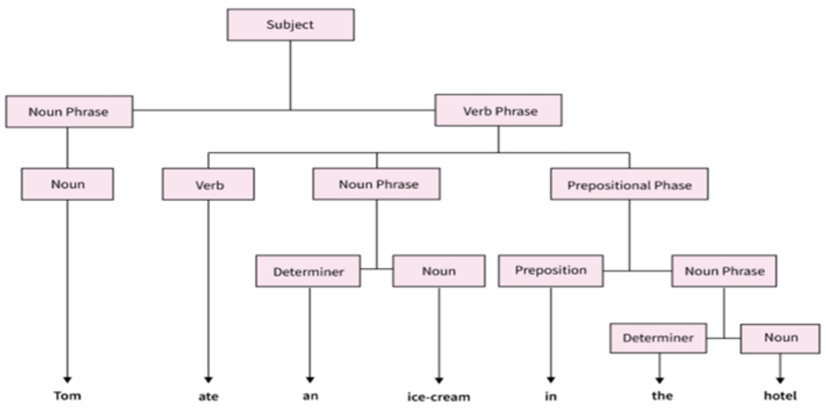
- Lexicons help identify specific types of entities like names, places, or dates.

Example:
Lexicon: *New York* → Location, *Google* → Organization

- lexicon helps resolve the meaning of words that have multiple interpretations.
- Lexicons can list valid words, helping identify misspelled or incorrect terms.

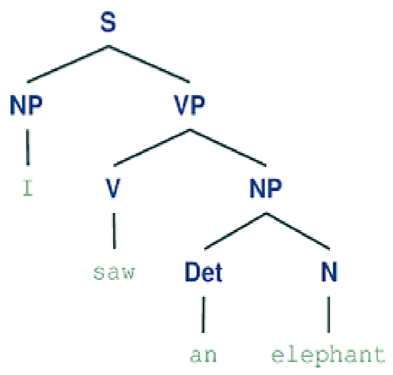
3. **PARSING**

Parsing in natural language processing (NLP) is the process of analyzing the grammatical structure of a sentence to understand its meaning. It involves breaking down a sentence into its constituent parts (such as words, phrases, and clauses) and identifying the relationships between these parts.



Types of Parsers:

i. Chart Parser: A chart parser is a tool used in computer programs to understand the structure of sentences. It helps computers figure out how words in a sentence fit together according to the rules of a language.



Example: Let us parse the sentence "the cat sleeps" with some simple rules:

A sentence (S) is made up of a noun phrase (NP) and a verb phrase (VP):

S → NP VP

A noun phrase (NP) can be a determiner (Det) followed by a noun (N):

NP → Det N

A verb phrase (VP) is just a verb (V): VP → V

The determiner "the" is a Det: Det → "the"

The word "cat" is a noun (N): N → "cat"

The word "sleeps" is a verb (V): V → "sleeps"

ii. Shift-reduce Parser

It is commonly used in compilers to analyse the syntax of programming languages. The main idea is to build the parse tree by shifting input symbols onto a stack and then reducing sequences of symbols to grammar rules. Example: Let us parse the arithmetic expression "id + id" using the following grammar:

E → E + E

E → id

Steps:

• Initialize:

Stack: [ ]

Input: [id, +, id]

• Shift id:

Stack: [id]

Input: [+, id]

• Reduce E → id:

Stack: [E]

Input: [+, id]

• Shift +:

Stack: [E, +]

Input: [id]

• Shift id:

Stack: [E, +, id]

Input: [ ]

• Reduce E → id:

Stack: [E, +, E]

Input: [ ]

Now the stack contains a single start symbol (E), and the input buffer is empty, so the parsing is successful.

3.1. <u>Dependency Parsing</u>

Dependency Parsing is a method used in Natural Language Processing (NLP) to analyze the grammatical structure of a sentence. It shows how words in a sentence are connected and which words depend on others to form meaning. This structure is usually represented as a dependency tree.

- Every word in a sentence has a relationship with another word.
- The main word in the sentence (usually the verb) is called the head.
- Other words are connected to the head as dependents.

For the sentence:
"The cat eats a fish"

Head: *eats* (the main verb).

Dependents:

cat depends on *eats* (subject).

fish depends on *eats* (object).

The and *a* depend on *cat* and *fish* respectively (determiners).

Dependency tree:

```
     Eats
     /   \
  cat    fish
   |        |
  The       a
```

4. Augmented Grammar

Augmented Grammar is an improved version of basic grammar that adds extra information to make it easier for computers to understand language better. It includes features like tense**,** number, and gender, which help handle complex sentences and improve tasks like translation and sentence analysis.

It makes grammatical rules more detailed.

Helps resolve ambiguity in sentences.

Captures specific details about words, like whether a noun is singular or plural or whether a verb is in the past or present tense.

Basic grammar only tells us how words fit together:

- Sentence → Noun Phrase + Verb Phrase
- Noun Phrase → Article + Noun
- Verb Phrase → Verb + Noun Phrase

For the sentence: *"The cat chases the mouse."* Basic Grammar only checks that the structure is correct.

In **augmented grammar**, we add extra features:

- Noun: {Singular/Plural, Gender}
- Verb: {Tense, Agreement with Subject}

For the same sentence: *"The cat chases the mouse."*

- *cat*: {Singular, Neutral}
- *chases*: {Present Tense, Matches Singular Subject}
- *mouse*: {Singular, Neutral}

This ensures the verb *chases* is correct for the singular subject *cat*. If you said *"The cat chase the mouse,"* augmented grammar would catch the error because the verb does not match the subject.

4.1. Semantic Interpretation

Semantic Interpretation is the process of understanding the meaning of words, sentences, or entire texts in Natural Language Processing (NLP). Unlike syntax, which focuses on the structure or grammar of a sentence, semantics deals with the meaning conveyed by the sentence. Semantic interpretation aims to map a sentence into a representation that captures its meaning, which can then be used for further tasks like answering questions, machine translation, or text summarization.

Semantics helps in understanding the individual meanings of words and how they contribute to the overall meaning of a sentence.

Example: *"bank"* can refer to a financial institution or the side of a river. Semantic interpretation helps determine the correct meaning based on context.

It helps in understanding how the sentence conveys meaning.

Example:

"The cat chased the mouse." - The subject *cat* is performing the action *chased* on the object *mouse*, and the sentence describes an event.

It solves the ambiguity in Sentences

Sentence: "I saw the man with a telescope."

There are two possible meanings:

a) I used a telescope to see the man.
b) The man I saw was holding a telescope.

5. Complications in Real Natural Language

Real natural language is full of complexities and challenges, which make it difficult for computers to understand and process it effectively.

i. Ambiguity occurs when a word, phrase, or sentence has more than one possible meaning.

Example: *"I saw the man with a telescope."*

It could mean "I used a telescope to see the man" or "The man I saw had a telescope."

Computers must figure out which meaning makes sense based on the context, which can be tricky.

a) Lexical ambiguity: When a word has more than one meaning. The word "bat" can mean a flying mammal or a piece of sports equipment.
b) Semantic ambiguity: the interpretation of a sentence in context. For Example: "I saw the boy on the beach with my binoculars". This could mean that I saw a boy through my binoculars or the boy had my binoculars with him.
c) Syntactic ambiguity: When a sentence has multiple meanings because of word combinations. Example: "Visiting relatives can be boring." This could mean that "the act of visiting relatives is boring" or "relatives who are visiting can be boring."

ii. Misspellings and Grammatical Errors: Overcoming Misspelling and Grammatical Error are the basic challenges in NLP, as there are different forms of linguistics noise that can impact accuracy of understanding and analysis.
iii. Diverse Grammar and Syntax: Different languages have different grammatical rules and sentence structures, making universal parsing difficult.
iv. Idiomatic Expressions: Idiomatic expressions are phrases that do not mean what the individual words suggest.

Example: *"Kick the bucket"* means "to die", not to literally kick a bucket.

Since these phrases do not follow standard grammar rules, it can be hard for a computer to interpret them correctly.

v. Spelling and Grammar Mistakes and grammar errors are common in real-world language, especially in casual writing like emails or social media posts. Computers must handle these mistakes and still understand the meaning.

6. Natural Language Tasks

Speech Recognition is a technology that converts spoken sound into written text. It involves analyzing audio signals to detect and transcribe words, enabling computers to understand and process human speech. The system typically uses algorithms and models that are trained to recognize different speech patterns, accents, and languages.

Text-to-Speech Synthesis refers to the process of converting written text into spoken language. The objective is to ensure that the pronunciation of each word is accurate, and that the speech sounds natural. This involves generating appropriate intonation, pauses, and emphasis, so the speech flows smoothly and is easily understood by the listener. It is widely used in voice assistants and accessibility tools for visually impaired individuals.

Machine Translation (MT) is a field of computational linguistics that focuses on automatically translating text from one language into another. Machine translation systems analyze the source text, identify its meaning, and generate an equivalent translation in the target language. The goal of MT is to break down language barriers and provide fast and efficient translations without the need for human translators.

Information Extraction is the process of identifying and extracting specific pieces of information from unstructured text. This includes detecting entities, such as names, dates, and locations, as well as relationships between those entities. Information extraction enables machines to extract structured data from large volumes of text, making it easier to analyze and organize information for further use.

Information Retrieval is the task of locating relevant documents or pieces of information based on a user’s query. This involves searching through large collections of documents, ranking them according to relevance, and presenting the most pertinent results. Information retrieval is fundamental to search engines, which help users find the most useful information from vast amounts of data available online.