Introduction to Artificial Intelligence

MODULE 03
Natural Language Processing

Department of Computer Science, YIASCM

Index:

- Introduction to NLP
- Basics of Syntactic Processing
- Meaning,concept of parser
- Types of Parsing, Concept of derivation, Types of derivation
- Concept of grammar, Phrase structure or Constituency Grammar
- Dependency Grammar, Context-Free Grammar
- Basic of Semantic analysis
- Basics of grammar free analyzers

What is NLP?

- NLP stands for Natural Language Processing, which is a part of Computer Science, Human language, and Artificial Intelligence.
- It is the technology that is used by machines to understand, analyse, manipulate, and interpret human's languages.
- It helps developers to organize knowledge for performing tasks such as translation, automatic summarization, Named Entity Recognition (NER), speech recognition, relationship extraction, and topic segmentation.

History of NLP

- **NLP Began in 1950**: Alan Turing introduced NLP in his paper "Computing Machinery and Intelligence.“
- **Based on Artificial Intelligence**: NLP focuses on automatically interpreting and generating human language.
- **Heuristics-Based NLP**: Early NLP used rule-based methods, like regular expressions, based on expert knowledge.
- **Machine Learning-Based NLP**: Later, NLP used machine learning to learn patterns from data, using algorithms like Naive Bayes, SVM (Support Vector Machine), and HMM (Hidden Markov Model).

History of NLP

- **Neural network-based NLP uses deep learning**, which offers high accuracy but needs a lot of data and time to train.
- It requires high computational power to train the model. Furthermore, it is based on neural network architecture. Examples: Recurrent neural networks (RNNs), Long short-term memory networks (LSTMs), Convolutional neural networks (CNNs), Transformers, etc.

Components of NLP

There are two components of NLP as given:

- **Natural Language Understanding (NLU):** Understanding involves the following tasks:
  - Mapping the given input in natural language into useful representations.
  - Analyzing different aspects of the language.
- **Natural Language Generation (NLG):** It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation.

Components of NLP

It involves:

- **Text planning**: It includes retrieving the relevant content from knowledge base.
- **Sentence planning**: It includes choosing required words, forming meaningful phrases, setting tone of the sentence.
- **Text Realization**: It is mapping sentence plan into sentence structure.

The NLU is harder than NLG.

Difficulties in NLU

- NL has an extremely rich form and structure.
- It is very ambiguous. There can be different levels of ambiguity:
  - **Lexical ambiguity**: It refers to the phenomenon where a single word has multiple meanings or interpretations, causing confusion in understanding. For example, the word "bank" can refer to a financial institution or the side of a river, depending on the context.

Difficulties in NLU

- **Syntax Level ambiguity:** It occurs when a sentence can be interpreted in more than one way due to its structure or word arrangement.

  For Example:

  "The chicken is ready to eat.“

  This sentence can have two meanings:

  - The chicken (bird) is hungry and ready to eat something.
  - The chicken (meal) is cooked and ready to be eaten.

Difficulties in NLU

- **Referential ambiguity** in Natural Language Understanding occurs when it is unclear which noun or entity a pronoun or phrase is referring to in a sentence.

  For Example:

  - **Sentence**: "John told Mark that he won the prize.“
  - **Ambiguity**: It's unclear whether "he" refers to John or Mark.

In this case, the pronoun "he" creates referential ambiguity because it can refer to more than one person.

NLP Terminology

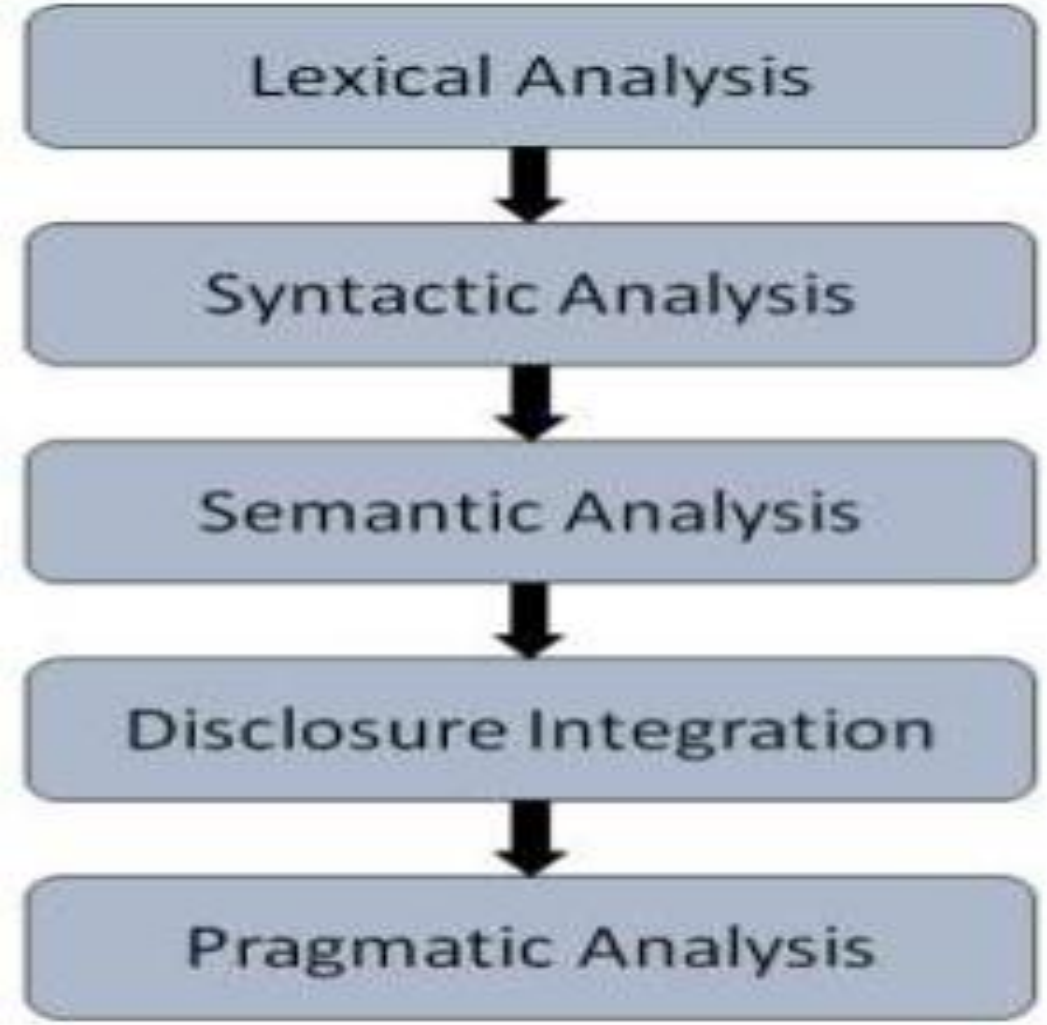
- **Phonology**: The study of the sounds in a language and how they are used.
- **Morphology**: The study of how words are formed from smaller units called morphemes.
- **Morpheme**: The smallest meaningful unit in a language (e.g., "un-" in "undo" or "-s" in "cats").
- **Syntax:** The set of rules that determine how words are arranged to form sentences.
- **Semantics:** The study of meaning in language, focusing on what words, phrases, and sentences mean.

NLP Terminology

- **Pragmatics**: It deals with how language is used in context, focusing on the meaning beyond the literal interpretation. It considers the speaker's intentions and how listeners understand them.
- **Discourse**: Discourse refers to language use in a broader context, such as conversations or written texts. It focuses on the structure and flow of communication beyond single sentences, looking at how ideas are connected.
- **World Knowledge**: World knowledge is the general information and understanding of how the world works that humans use to interpret language. It helps in understanding the meaning of words and sentences based on real-world facts.

Steps in NLP

There are general five steps:



Steps in NLP

There are general five steps:

1. **Lexical Analysis: Lexical Analysis** in NLP is the process of breaking down text into individual words or tokens, and understanding their basic meaning and structure.

   For example:

   - In the sentence "**The cat sat on the mat**":
   - **Tokens**: ["The", "cat", "sat", "on", "the", "mat"]
   - **Lexical Analysis**: Identifies "The" and "the" as articles, "cat" and "mat" as nouns, and "sat" and "on" as verbs and prepositions, respectively.

Steps in NLP

There are general five steps:

2. **Syntactic Analysis (Parsing):** It involves breaking down a sentence into its parts (such as subjects, verbs, and objects) and determining how these parts are connected according to grammar rules.

For Example:

- For the sentence "The cat sat on the mat":
- **Syntactic Analysis** will identify "The cat" as the subject, "sat" as the verb, and "on the mat" as the prepositional phrase describing where the cat sat.
- It helps in understanding that "The cat" performs the action of "sat," and "on the mat" provides additional information about the location of the action.

Steps in NLP

There are general five steps:

3. **Semantic Analysis:** It is the process of understanding the meaning of words and sentences in context. It involves interpreting the meanings and relationships between words to understand the overall message.

   For example:

   - If the sentence is "The bank was closed," semantic analysis helps determine whether "bank" refers to a financial institution or the side of a river, based on the context.

Steps in NLP

4. **Discourse Integration** refers to how language processing systems handle the relationships and connections between different parts of a text or conversation to understand its overall meaning.

   For Example:

   - In the sentence, “John went to the store. He bought some milk,” discourse integration helps recognize that “He” refers to “John,” even though “John” is mentioned in a separate sentence.

Steps in NLP

5. **Pragmatic Analysis** refers to understanding the context and intended meaning behind language use in real-life situations. It focuses on how language is used to convey meaning beyond the literal interpretation, considering factors like the speaker's intent, the situation, and conversational context.

   For Example:

   - In the sentence "Can you pass the salt?" the literal meaning is a question about the ability to pass the salt. However, pragmatically, it is understood as a polite request for someone to pass the salt during a meal.

Activity 10

1. What is Natural Language Processing (NLP), and how does it differ from traditional programming languages? Provide a brief overview of NLP techniques used for language understanding.
2. Why is NLP important in today's digital age? Explain examples of NLP applications in everyday life.
3. How has NLP evolved over the years? Briefly explain key milestones in the history of NLP, from early rule-based systems to modern machine learning approaches.

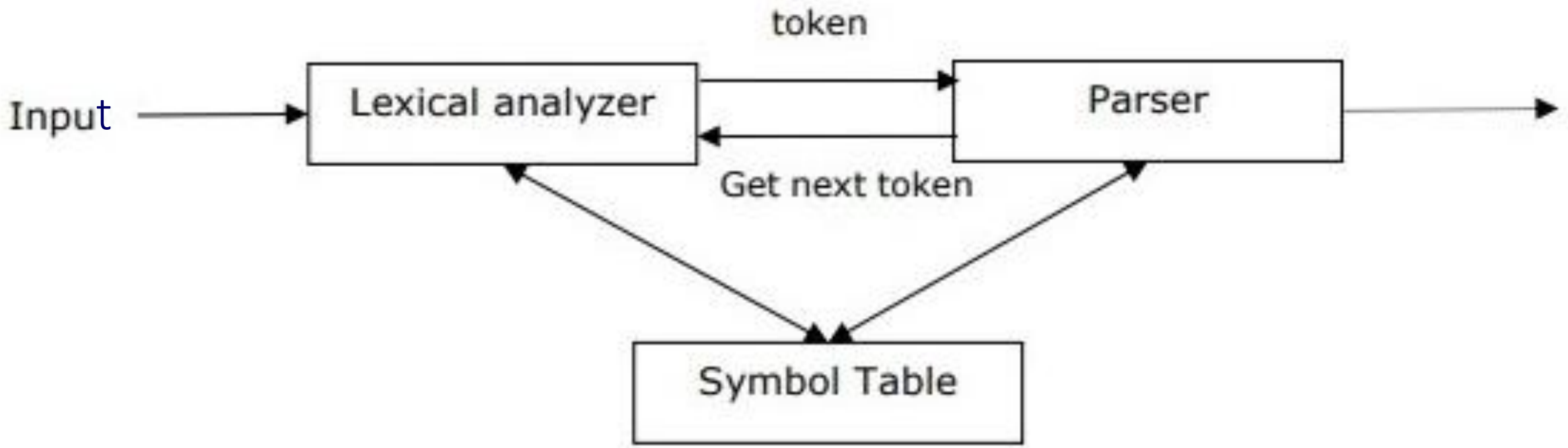
Syntactic analysis

- Syntactic analysis or parsing or syntax analysis is the third phase of NLP.
- The purpose of this phase is to draw exact meaning, or you can say dictionary meaning from the text.
- Syntax analysis checks the text for meaningfulness comparing to the rules of formal grammar. For example, the sentence like “hot ice-cream” would be rejected by semantic analyzer.
- In this sense, syntactic analysis or parsing may be defined as the process of analyzing the strings in natural language conforming to the rules of formal grammar.
- The origin of the word **‘parsing’** is from Latin word **‘pars’** which means **‘part’**.

Concept of Parser

- **Parsing** is the process of analyzing a sequence of symbols (such as words, tokens, or code) to determine its grammatical structure according to a set of rules or a formal grammar. It is commonly used in programming language processing, natural language processing, and data structure manipulations.
- It is used to implement the task of parsing. It may be defined as the software component designed for taking input data (text) and giving structural representation of the input after checking for correct syntax as per formal grammar.
- It also builds a data structure generally in the form of parse tree or abstract syntax tree or other hierarchical structure.

Concept of Parser



Concept of Parser

The main roles of the parse include −

- To report any syntax error.
- To recover from commonly occurring error so that the processing of the remainder of program can be continued.
- To create parse tree.
- To create symbol table.
- To produce intermediate representations (IR).

Parsing

- Parsing is a crucial technique in Natural Language Processing (NLP) that involves analyzing a string of symbols, either in natural language or computer languages, according to the rules of a formal grammar.
- Parsing helps in understanding the syntactic structure of a sentence and is essential for various NLP tasks such as machine translation, question answering, and text-to-speech systems.

Types of Parsing in NLP

Syntactic Parsing (Syntax Analysis)

- **Purpose:** Determines the **structure** of a sentence or code based on grammar rules.
- **Focus:** Focuses purely on the **form** and grammatical correctness without concern for meaning.
- **Output**: A parse tree or syntax tree that represents the syntactic structure according to a given grammar.

Types of Parsing in NLP

There are two main approaches: **Constituency Parsing** and **Dependency Parsing**.

1. Constituency Parsing

Constituency Parsing breaks down a sentence into its parts, or "constituents," like noun phrases and verb phrases. It creates a parse tree that shows how these parts fit together to form the sentence's hierarchical structure.

2. Dependency Parsing

Dependency Parsing focuses on the grammatical relationships between words. It constructs a tree where each word is connected to another word, indicating a dependency relationship, such as subject-verb-object.

Types of Parsing in NLP

Semantic Parsing

- **Purpose:** Extracts the **meaning** from a sentence or code by interpreting its structure.
- **Focus:** Focuses on what the sentence **means** rather than just how it is structured.
- **Output**: A logical form or some representation of the sentence's meaning.

Types of Parsing in NLP

Key Points of Semantic Parsing

1. **Extracting Meaning**: Unlike syntactic parsing, which looks at the structure, semantic parsing tries to understand what the sentence means.
2. **Context Understanding**: It analyzes the roles of words in a specific context and how they relate to each other.
3. **Applications**:
    - **Question Answering**: Helps in understanding and answering questions correctly.
    - **Knowledge Base Populating**: Extracts information from text to fill in databases.
    - **Text Understanding**: Helps in comprehending the overall meaning of documents and texts.

Activity 11

Identify the syntactic components of the given sentences by breaking it down into its basic parts of speech, such as noun phrases (NP), verb phrases (VP), adjectives (Adj), prepositional phrases (PP), etc. Explain the semantic interpretation by describing the meaning conveyed by the sentence, focusing on the roles of the subject, predicate, and any additional descriptive elements.

1. The intelligent student solved the complex problem quickly.
2. The small child happily played with the colorful toys.
3. The dedicated team worked on the challenging project throughout the night.

Parsing Techniques in NLP

The fundamental link between a sentence and its grammar is derived from a parse tree. A parse tree is a tree that defines how the grammar was utilized to construct the sentence. There are mainly two parsing techniques, commonly known as top-down and bottom-up.

Parsing Techniques in NLP

YENEPOYA
(DEEMED TO BE UNIVERSITY)

1. Top-Down Parsing

- **Top-down parsing** is a method of parsing where the process starts from the **root** of the parse tree (usually the start symbol of a grammar) and tries to break down the sentence into smaller components (like words and phrases), following the grammar rules. The parser **predicts** what structure the sentence might have and tries to match it with the actual input.
- This type of parsing uses a recursive approach, meaning it keeps breaking down the sentence until it matches the given input or finds that no valid parse exists.

Parsing Techniques in NLP

How Top-Down Parsing Works:

- **Start with the start symbol** (e.g., Sentence or S).
- **Apply grammar rules** to expand the start symbol into smaller symbols (e.g., phrases, words).
- **Compare** these predicted symbols with the actual words in the input sentence.
- **Recursively apply** the rules to break down each component until the entire sentence is parsed or no valid rule can be applied.

Parsing Techniques in NLP

YENEPOYA
(DEEMED TO BE UNIVERSITY)

Example

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

NP → Det N (A noun phrase is a determiner followed by a noun)

VP → V NP (A verb phrase is a verb followed by a noun phrase)

Det → 'the‘

N → 'cat' | 'mouse‘

V → 'chases‘

And the sentence to parse: **"the cat chases the mouse"**.

Parsing Techniques in NLP

Step-by-Step Top-Down Parsing

1. Start with the start symbol (S):

- The sentence structure is predicted as **S → NP VP**.

2. Expand NP (Noun Phrase):

- **NP → Det N**.
- Now, predict that the first word in the sentence will be a **Det** (determiner) followed by a **N** (noun).

Parsing Techniques in NLP

Step-by-Step Top-Down Parsing

3. Match with input:

- The input starts with "the", which matches the rule **Det → 'the'**.
- Now, look for a noun. The next word in the input is "cat", which matches **N → 'cat'**.

4. Expand VP (Verb Phrase):

- **VP → V NP**.
- The verb phrase is predicted as a verb followed by a noun phrase.

Parsing Techniques in NLP

Step-by-Step Top-Down Parsing

5. Match with input:

- The next word in the input is "chases", which matches **V → 'chases'**.
- Now, look for another noun phrase (NP).

6. Expand NP:

- **NP → Det N** again.
- The next word is "the", matching **Det → 'the'**, and the last word is "mouse", matching **N → 'mouse'**.

Parsing Techniques in NLP

YENEPOYA
(DEEMED TO BE UNIVERSITY)

Step-by-Step Top-Down Parsing

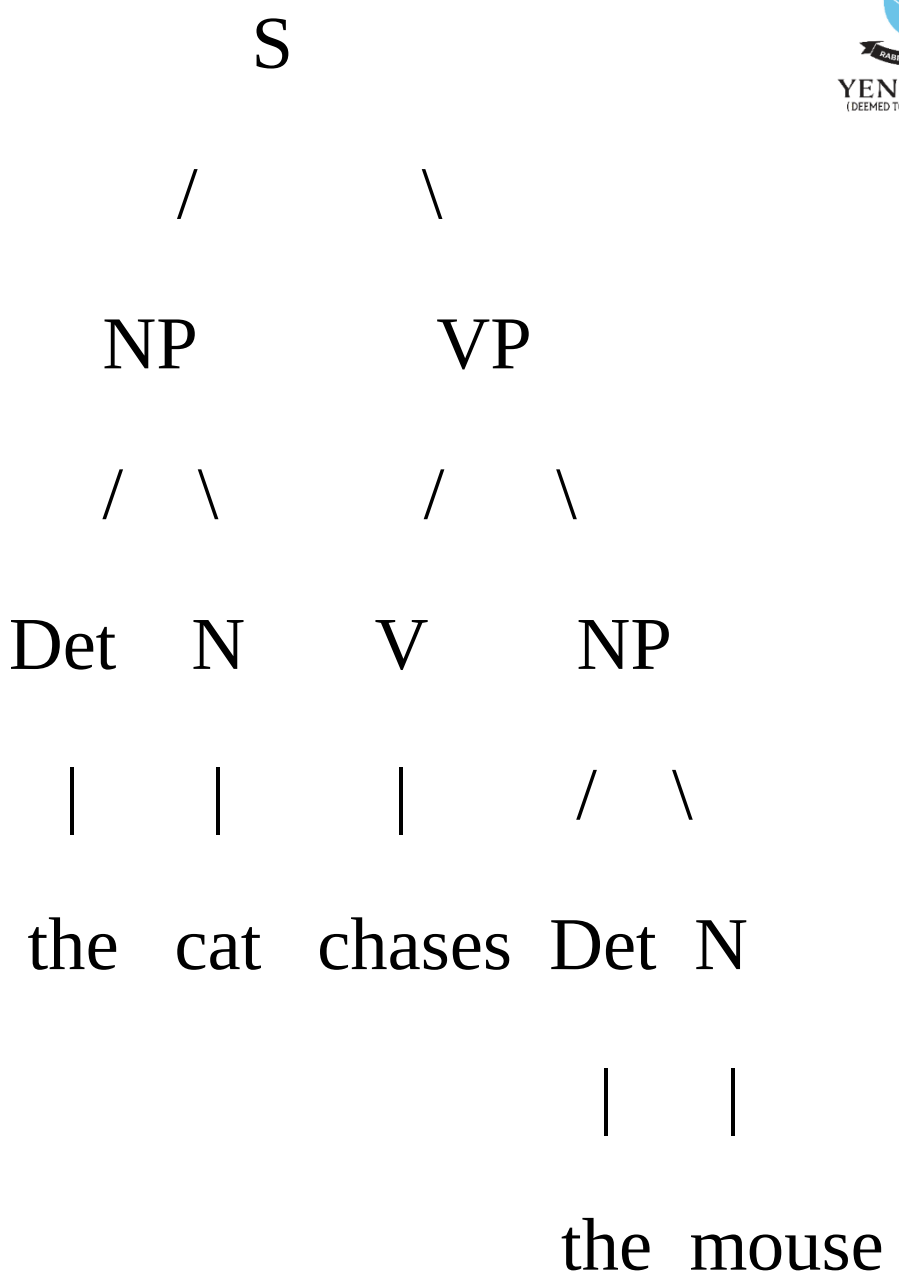
7. Complete the parse:

- The whole input "the cat chases the mouse" is successfully parsed according to the grammar.

Parsing Techniques in NLP

Summary of Parse Tree:

In top-down parsing, the parser **starts with a prediction** (grammar rules) and checks if the input sentence matches that prediction, recursively breaking down the sentence



Parsing Techniques in NLP

2. Bottom-Up Parsing

Bottom-Up Parsing is a parsing technique in which the parser starts from the input (the individual words or tokens of a sentence) and tries to build the parse tree by working its way **upwards**, combining smaller parts (like words or phrases) into larger structures until it reaches the start symbol of the grammar.

Parsing Techniques in NLP

How Bottom-up Parsing Works:

- The parser **begins with the input words** and repeatedly tries to find grammar rules that can combine them into bigger phrases.
- **Reduces** small parts (like words) into larger parts (like noun phrases or verb phrases) based on grammar rules.
- Continues this process until the whole sentence is reduced to the **start symbol** of the grammar (e.g., a "sentence").

Parsing Techniques in NLP

Example:

Consider the sentence:

"The dog barks.“

Grammar Rules:

- **S → NP VP** (A sentence is made of a noun phrase followed by a verb phrase)
- **NP → Det N** (A noun phrase is made of a determiner and a noun)
- **VP → V** (A verb phrase is just a verb)
- **Det → 'The'**
- **N → 'dog'**
- **V → 'barks'**

Parsing Techniques in NLP

Example:

Step-by-Step Process for bottom up parsing:

1. Input: "The dog barks.“

2. Look at the words: Identify parts of speech (Det = "The", N = "dog", V = "barks").

- Det: "The“
- N: "dog“
- V: "barks"

Parsing Techniques in NLP

Example:

Step-by-Step Process:

3. Apply the rules bottom-up:

- First, combine **Det + N → NP**:
  - "The" + "dog" → NP ("The dog")
- Now combine **V → VP**:
  - "barks" → VP

Parsing Techniques in NLP

Example:

Step-by-Step Process:

4. Final reduction:

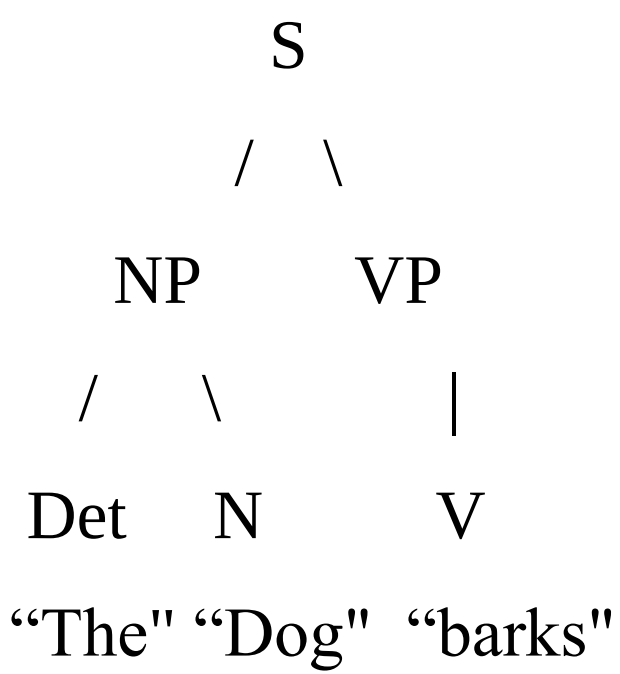
- Combine **NP + VP → S** (Sentence):
  - "The dog" + "barks" → S (The dog barks)

The parser successfully constructs a valid parse tree for the sentence, starting from the individual words and working its way up to form the full sentence.

Parsing Techniques in NLP

Bottom-Up Parsing

The parse tree looks like this:



How Does the Parser Work?

- The first step is to identify the sentence's subject. The parser divides the text sequence into a group of words that are associated with a phrase. So, this is the collection of words that are related to one another is referred to as the subject.
- The sentence means that syntactic parsing and parts of speech focus on how words are arranged in a sentence, following specific rules, without depending on the meaning or context of the sentence.
- The most important thing to remember is that grammar is always syntactically valid, even if it may not make contextual sense.

Applications of Parsing in NLP

- **Syntactic Analysis:** SA Parsing helps in determining the syntactic structure of sentences by detecting parts of speech, phrases, and grammatical relationships between words. This information is critical for understanding sentence grammar.
- **Named Entity Recognition (NER):** NER parsers can detect and classify entities in text, such as people, organizations, and location's names, among other things. This is essential for information extraction and text comprehension.
- **Semantic Role Labeling (SRL):** SRL parsers determine the semantic roles of words in a sentence, such as who is the "agent," "patient," or "instrument" in a given activity. It is essential for understanding the meaning of sentences.

Applications of Parsing in NLP

- **Machine Translation**: Parsing can be used to assess source language syntax and generate syntactically correct translations in the target language. This is necessary for machine translation systems such as Google Translate.
- **Question Answering**: Parsing is used in question-answering systems to help break down a question into its grammatical components, allowing the system to search a corpus for relevant replies.
    - In the context of question-answering systems, a corpus is a database or set of documents that the system searches through to find relevant answers.

Applications of Parsing in NLP

- **Text Summarization**: TS Parsing is the process of extracting the essential syntactic and semantic structures of a text, which is necessary for producing short and coherent summaries.
- **Information Extraction**: IE Parsing is used to extract structured information from unstructured text, such as data from resumes, news articles, or product reviews.

Activity 12

Analyze the following sentences using top down and bottom up parsing techniques.

1. The cat is on the mat.

2. The dog barked loudly.

[Top down parsing: Provide a step-by-step breakdown of how the sentence can be parsed starting from the start symbol (usually the sentence, S) and applying production rules until you reach the terminal symbols (words of the sentence).

Bottom Up parsing: Provide a step-by-step breakdown of how the sentence can be parsed starting from the terminal symbols (words of the sentence) and applying production rules in reverse until you reach the start symbol (S)]

Concept of Derivation

- In order to get the input string, we need a sequence of production rules. Derivation is a set of production rules.
- In the context of parsing, a **non-terminal** is a symbol used in grammar rules that can be further expanded or replaced by other symbols (both terminals and non-terminals). Non-terminals represent abstract grammatical categories or structures, such as a sentence (S), noun phrase (NP), or verb phrase (VP).
- During parsing, the non-terminal symbols are progressively replaced by other symbols based on grammar rules (called production rules) until only terminal symbols, which are the actual words or tokens in the sentence, remain. Non-terminals help define the structure of a sentence but aren't the final words themselves.

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

1. Grammar Rules (Production Rules):

We'll define some basic production rules for parsing this sentence:

- **S** → NP VP (A sentence can be a noun phrase followed by a verb phrase)
- **NP** → Det N (A noun phrase can be a determiner followed by a noun)
- **VP** → V (A verb phrase can just be a verb)
- **Det** → "The" (The determiner "The")
- **N** → "cat" (The noun "cat")
- **V** → "sleeps" (The verb "sleeps")

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

2. Start with the Non-terminal S (Sentence):

- **S** is the starting non-terminal that represents the entire sentence.
- According to the rule:

  S → NP VP

  So, we break it down into a noun phrase (NP) and a verb phrase (VP).

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

3. Replace NP (Noun Phrase):

- Now, we expand **NP** using the rule:

  NP → Det N

  This breaks the noun phrase into a determiner (**Det**) and a noun (**N**).

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

4. Replace Det and N (Terminal symbols):

- **Det** → "The"

  We replace the non-terminal **Det** with the terminal word **"The"**.

- **N** → "cat"

  We replace the non-terminal **N** with the terminal word **"cat"**.

So now, the NP is fully replaced by **"The cat"**.

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

5. Replace VP (Verb Phrase)

- Now, we move to the verb phrase **VP** and use the rule:

  VP → V

  This means the verb phrase is replaced by a verb **V**.

Concept of Derivation: Example

Sentence: **"The cat sleeps.“**

Step-by-Step Parsing Process (Non-terminals and Terminals):

6. Replace V (Terminal symbol):

- **V** → "sleeps"

  We replace the non-terminal **V** with the terminal word **"sleeps"**.

Final Breakdown of the Sentence:

S → NP VP

NP → Det N → "The cat“

VP → V → "sleeps“

Thus, **"The cat sleeps."** is generated through the replacement of non-terminal symbols by terminals following the production rules.

Concept of Derivation: Example

YENEPOYA
(DEEMED TO BE UNIVERSITY)

Non-terminals:

These are the abstract symbols that represent parts of the sentence and need to be replaced:

- **S** (Sentence)
- **NP** (Noun Phrase)
- **VP** (Verb Phrase)
- **Det** (Determiner)
- **N** (Noun)
- **V** (Verb)

Terminals:

These are the actual words in the sentence, which are the final outputs after replacing non-terminals:

- **"The"** (Determiner)
- **"cat"** (Noun)
- **"sleeps"** (Verb)

Types of Derivation

- **Left-most Derivation**

  In the left-most derivation, the sentential form of an input is scanned and replaced from the left to the right. The sentential form in this case is called the left-sentential form.

- **Right-most Derivation**

  In the right-most derivation, the sentential form of an input is scanned and replaced from right to the left. The sentential form in this case is called the right-sentential form.

Types of Derivation

- **Left-most Derivation: Example**

  Sentence: “**The cat sleeps**”

  Grammar Rules:

  - **S** → NP VP (A sentence is a noun phrase followed by a verb phrase)
  - **NP** → Det N (A noun phrase is a determiner followed by a noun)
  - **VP** → V (A verb phrase is a verb)
  - **Det** → "The"
  - **N** → "cat"
  - **V** → "sleeps"

Types of Derivation

Left-most Derivation: Example

In left-most derivation, we always expand the **left-most non-terminal** first.

Step-by-step process:

- **S** → NP VP
  We start with the left-most non-terminal **S** and expand it to **NP** and **VP**.
- **NP** → Det N
  Now, we expand the left-most non-terminal **NP** into **Det** and **N**.
- **Det** → "The"
  The left-most non-terminal is now **Det**, so we replace it with the terminal word **"The"**.
- **N** → "cat"
  Now the left-most non-terminal is **N**, so we replace it with **"cat"**.
- **VP** → V
  Finally, we expand the remaining **VP** to **V**.
- **V** → "sleeps"
  We replace the **V** with the terminal word **"sleeps"**.

Types of Derivation

Left-most Derivation: Example

Left-most derivation sequence:

- **S** → NP VP
- **NP VP** → Det N VP
- **Det N VP** → "The" N VP
- **"The" N VP** → "The" "cat" VP
- **"The cat" VP** → "The cat" V
- **"The cat" V** → "The cat sleeps"

Thus, in the left-most derivation, we expand non-terminals from left to right.

Types of Derivation

Right-most Derivation: Example

In right-most derivation, we always expand the **right-most non-terminal** first.

Step-by-step process:

- **S** → NP VP
  Start with **S** and expand it to **NP** and **VP**.
- **VP** → V
  Instead of expanding **NP** (left-most), we expand the **right-most** non-terminal, which is **VP** into **V**.
- **V** → "sleeps"
  We replace the **V** with the terminal word **"sleeps"**.
- **NP** → Det N
  Now, we go back to the left and expand **NP** into **Det** and **N**.
- **N** → "cat"
  We replace the **right-most** non-terminal **N** with the terminal word **"cat"**.
- **Det** → "The"
  Finally, we replace **Det** with **"The"**.

Types of Derivation

Right-most Derivation: Example

Right-most derivation sequence:

- **S** → NP VP
- **NP VP** → NP V
- **NP V** → NP "sleeps"
- **NP "sleeps"** → Det N "sleeps"
- **Det N "sleeps"** → Det "cat" "sleeps"
- **Det "cat" "sleeps"** → "The" "cat" "sleeps"
- In right-most derivation, non-terminals are expanded from right to left.

Concept of Grammar

Grammar is essential for defining the syntactic structure of well-formed sentences in both natural languages (like English, Hindi) and programming languages (like C). It ensures that communication, whether between humans or between humans and computers, is clear and meaningful.

Natural Language Grammar

Purpose: Defines rules for combining words to form meaningful sentences.
Example:

- Sentence: "The cat sat on the mat."
- Rule: [Determiner] + [Noun] + [Verb] + [Preposition] + [Determiner] + [Noun]

Concept of Grammar

Formal Language Grammar (Programming)

Purpose: Defines rules for combining symbols to form valid programs.

Example:

Function in C:

```
int add(int a, int b) {
    return a + b;
}
```

Rule (Syntax): return_type function_name(parameter_list)

{ body }

Concept of Grammar

Chomsky's Model of Grammar (1956)

Noam Chomsky introduced a mathematical model for grammars, which is useful for describing both natural and programming languages.

Formal Grammar: A 4 – Tuple Model

A grammar G can be formally represented as a 4-tuple (N,T,S,P):

1. **N or VN (Non-terminal symbols)**: Variables representing groups of symbols.
2. **T or $\sum$ (Terminal symbols)**: Basic symbols from which strings are formed.
3. **S (Start symbol)**: The initial non-terminal symbol, where S∈NS \in NS∈N.
4. **P (Production rules)**: Rules for transforming non-terminal symbols into other non-terminal or terminal symbols. The form is $\alpha \rightarrow \beta$, where α and β are strings composed of symbols from $N \cup \sum$, and α contains at least one non-terminal symbol.

Concept of Grammar

Example of a Formal Grammar

For a simple arithmetic expression grammar:

1. **N**: {Expression, Term, Factor}
2. **T**: {+, *, (,), id}
3. **S**: Expression
4. **P**:
    - Expression → Expression + Term | Term
    - Term → Term * Factor | Factor
    - Factor → (Expression) | id

Phrase Structure or Constituency Grammar

Phrase Structure Grammar, also known as Constituency Grammar, was introduced by Noam Chomsky. This type of grammar focuses on how sentences are composed of smaller units called constituents, which can be nested within each other.

Key Points

1. **Constituency Relation**:
   - Sentences are structured based on the relationship between constituents.
   - Derived from the subject-predicate structure in Latin and Greek grammar.
2. **Basic Clause Structure**:
   - Consists of a Noun Phrase (NP) and a Verb Phrase (VP).

Phrase Structure or Constituency Grammar

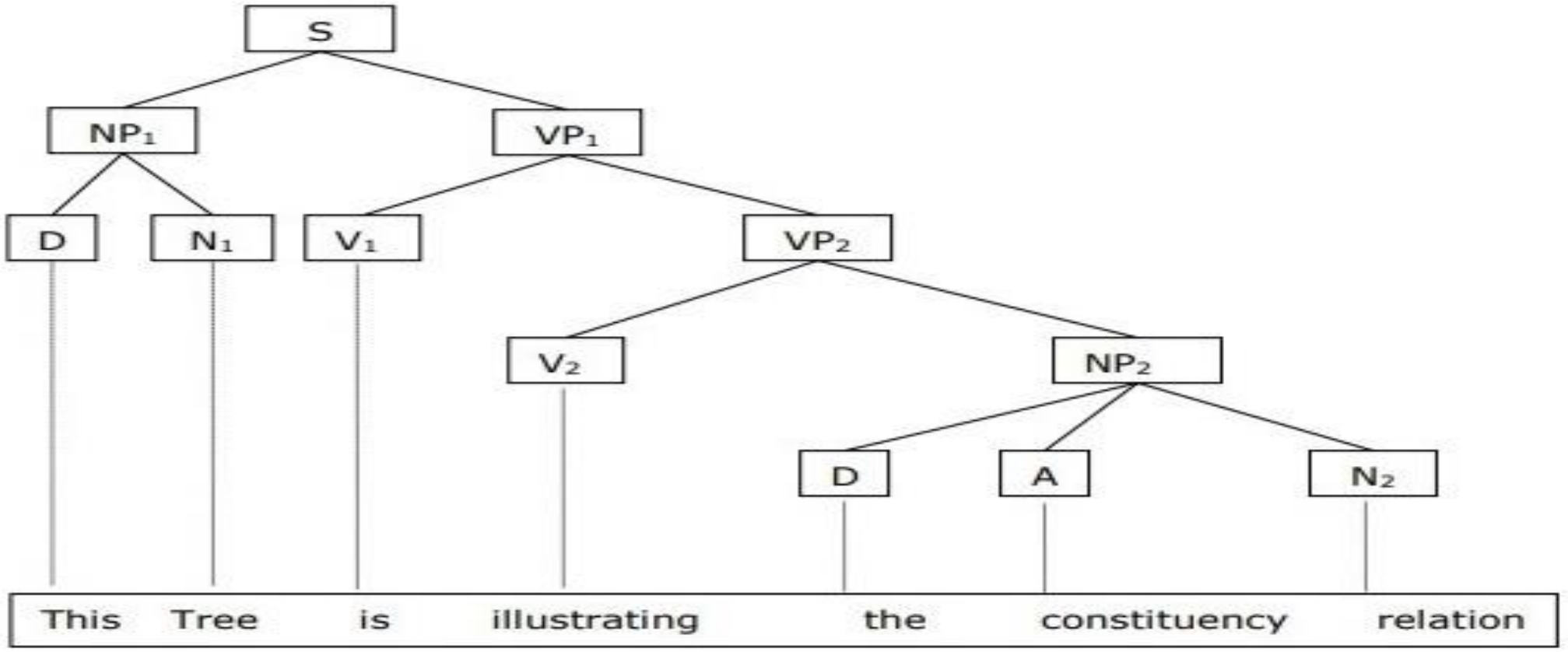
Example of Constituency Grammar

Let's break down the sentence: "This tree is illustrating the constituency relation."

Constituency Structure

1. **Sentence (S)**: The entire sentence.
2. **Noun Phrase (NP)**: The subject of the sentence.
    - Example: "This tree"
3. **Verb Phrase (VP)**: The predicate of the sentence, which includes the verb and additional components.
    - Example: "is illustrating the constituency relation"

Phrase Structure or Constituency Grammar



Dependency Grammar

Dependency Grammar (DG) is a type of syntactic structure that differs from constituency grammar. It focuses on the dependency relation between words in a sentence rather than on hierarchical phrases. DG was introduced by Lucien Tesnière and is characterized by the following points:

- **Linguistic Units and Directed Links**: In DG, words (linguistic units) are connected to each other by directed links that denote dependencies.
- **Central Role of the Verb**: The verb is the central element of the clause structure.
- **Dependencies**: Other syntactic units are connected to the verb and each other through directed links, forming dependencies.

Dependency Grammar

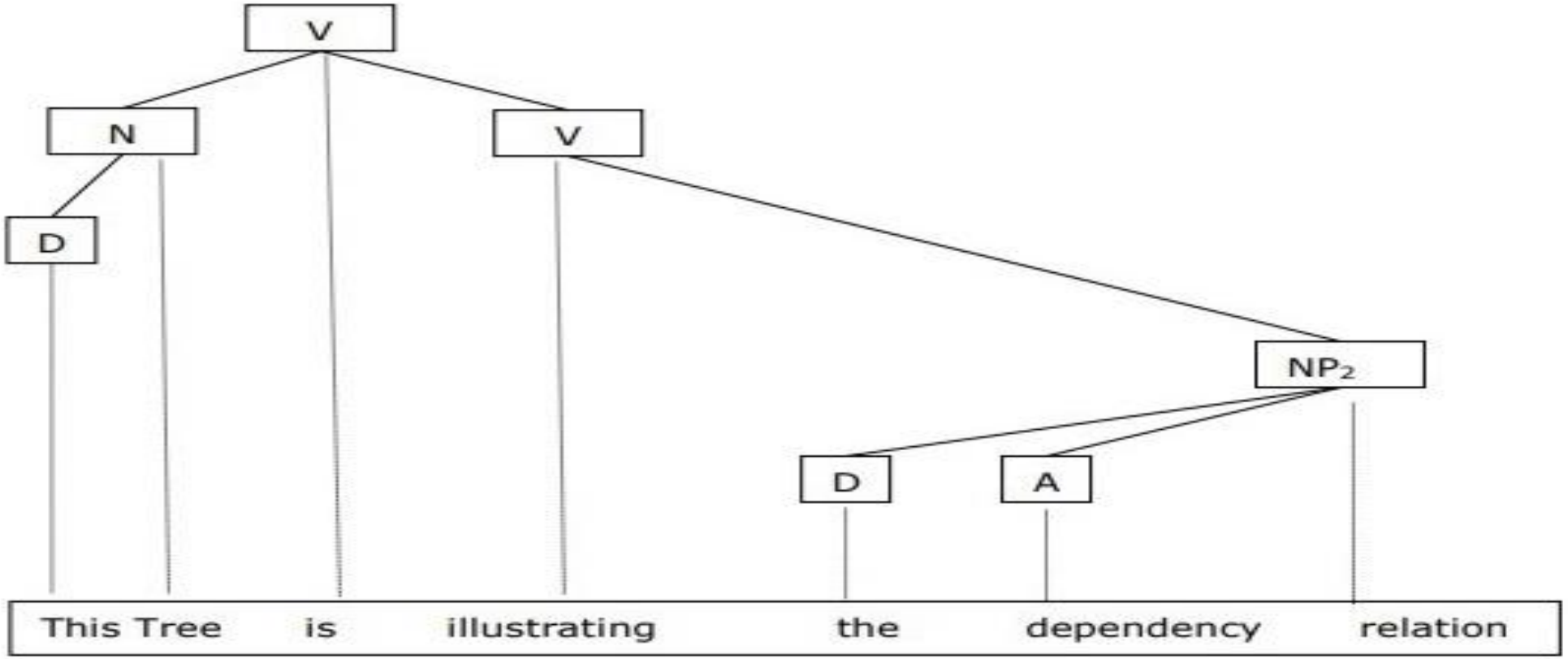
Example Sentence: "This tree is illustrating the dependency relation"

To understand how dependency grammar works, let's break down the sentence "This tree is illustrating the dependency relation" into its dependencies.

1. **Root Verb**: The main verb "illustrating" is the center of the clause.
2. **Subject**: "tree" is the subject of the verb "illustrating".
3. **Determiner**: "This" modifies "tree".
4. **Auxiliary Verb**: "is" is an auxiliary verb that helps the main verb "illustrating".
5. **Object**: "relation" is the object of "illustrating".
6. **Modifier**: "dependency" modifies "relation".

Dependency Grammar

We can write the sentence **“This tree is illustrating the dependency relation”** as follows;



Context-Free Grammar (CFG)

Context-Free Grammar (CFG), also known as CFG, is a formalism used to describe languages. It is more powerful than regular grammar and is capable of generating more complex languages. CFG is composed of four main components:

1. **Set of Non-terminals (V)**
2. **Set of Terminals (Σ)**
3. **Set of Productions (P)**
4. **Start Symbol (S)**

Context-Free Grammar (CFG)

Let's break down each component:

1. **Set of Non-terminals (V)**
   - Non-terminals are syntactic variables that represent sets of strings.
   - They help define the structure of the language generated by the grammar.
   - Example: ‘V = {S, A, B}’
2. **Set of Terminals (Σ)**
   - Terminals are the basic symbols from which strings are formed.
   - They are also called tokens.
   - Example: ‘Σ = {a, b}’

Context-Free Grammar (CFG)

YENEPOYA
(DEEMED TO BE UNIVERSITY)

Set of Productions (P)

- Productions define how terminals and non-terminals can be combined.
- Each production consists of a non-terminal, an arrow (→), and a sequence of terminals and/or non-terminals.
- Non-terminal symbols on the left side are rewritten as the sequence on the right side.
- Example: ‘$P = \{S \rightarrow AB, A \rightarrow a, B \rightarrow b\}$’

Start Symbol (S)

- The start symbol is a special non-terminal from which the production begins.
- Example: ‘$S = S$’

Context-Free Grammar (CFG)

Example CFG

Let's consider a simple example of a CFG that generates the language consisting of the string "ab":

- **Non-terminals (V)**: ‘V = {S, A, B}’
- **Terminals (Σ)**: ‘Σ = {a, b}’
- **Productions (P)**:
  - ‘S → AB’
  - ‘A → a’
  - ‘B → b’
- **Start Symbol (S)**: ‘S = S’

Context-Free Grammar (CFG)

Derivation Process

The derivation process shows how the start symbol is transformed into a string using the productions.

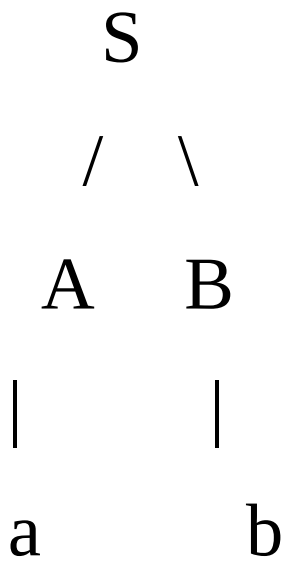
1. **Start with S**: ‘S’
2. **Apply S → AB**: ‘AB’
3. **Apply A → a**: ‘aB’
4. **Apply B → b**: ‘ab’

So, the string "ab" is generated by this CFG.

Context-Free Grammar (CFG)

Diagram

Here's a simple diagram representing the example CFG:



- ‘S’ is the start symbol.
- ‘S’ produces ‘A’ and ‘B’.
- ‘A’ produces the terminal ‘a’.
- ‘B’ produces the terminal ‘b’.

Semantic Analysis

Semantic analysis is the process of extracting meaning from text. It helps computers understand and interpret sentences, paragraphs, or entire documents by analyzing their structure and identifying relationships between words.

Key Points of Semantic Analysis

1. **Understanding Relationships**: It examines how words in a text relate to each other.
2. **Applications**: Used in tools like chatbots, search engines, and text analysis.
3. **Benefits**: Helps companies extract valuable information from unstructured data like emails and customer feedback.

Semantic Analysis

How Semantic Analysis Works

Lexical Semantics

- **Hyponyms and Hypernyms**: A hyponym is a specific instance of a general concept (hypernym). E.g., "orange" (hyponym) is a type of "fruit" (hypernym).
- **Meronomy**: Describes parts of a whole. E.g., a "segment" of an orange.
- **Polysemy**: Words that have multiple meanings related by a common core. E.g., "paper" can mean a material to write on or an academic article.
- **Synonyms**: Words with similar meanings. E.g., happy, content, ecstatic.
- **Antonyms**: Words with opposite meanings. E.g., happy and sad.
- **Homonyms**: Words that sound and are spelled the same but have different meanings. E.g., "orange" (color) and "orange" (fruit).

Semantic Analysis

Machine Learning in Semantic Analysis

- **Training Algorithms**: Feeding algorithms with text samples to learn from past observations.
- **Sub-tasks**:
  - **Word Sense Disambiguation**: Identifying the correct meaning of a word based on context. E.g., "date" can mean a day, a fruit, or a meeting.
  - **Relationship Extraction**: Detecting relationships between entities in a text. E.g., “Steve Jobs is one of the founders of Apple.”

Semantic Analysis

Semantic Analysis Techniques

Semantic Classification Models

1. **Topic Classification**: Categorizing text based on content.
    - E.g., Classifying support tickets as "Payment issue" or "Shipping problem".
2. **Sentiment Analysis**: Detecting emotions (positive, negative, or neutral) in text.
    - E.g., Tagging Twitter mentions by sentiment to gauge customer feelings.
3. **Intent Classification**: Determining what customers want to do next.
    - E.g., Tagging sales emails as “Interested” or “Not Interested”..

Semantic Analysis

Semantic Extraction Models

1. **Keyword Extraction**: Identifying important words and phrases in text.
    - E.g., Analyzing keywords in negative tweets to find common complaints.
2. **Entity Extraction**: Identifying names of people, companies, places, etc.
    - E.g., Extracting product names, shipping numbers, and emails from support tickets.

Basics of Grammar-Free Analyzers

Grammar-free analyzers in Natural Language Processing (NLP) are systems that process text without relying on predefined grammatical rules or formal grammars. Instead, they use statistical methods, machine learning algorithms, or other heuristics to analyze and understand language. Here are the key concepts and methods behind grammar-free analyzers:

Basics of Grammar-Free Analyzers

Key Concepts

1. **Statistical Methods**:
   - Rely on large corpora of text to understand language patterns.
   - Use frequency counts, probabilities, and co-occurrence statistics to analyze text.
2. **Machine Learning**:
   - Employ algorithms that learn from annotated data to make predictions or classifications.
   - Common approaches include supervised, unsupervised, and reinforcement learning.
3. **Heuristic Methods**:
   - Use rule-of-thumb strategies to process text.
   - These methods might include pattern matching, keyword extraction, or context analysis.

Basics of Grammar-Free Analyzers

YENEPOYA
(DEEMED TO BE UNIVERSITY)

Common Techniques

1. n-grams

Concept: An n-gram is a contiguous sequence of n items (usually words or characters) from a given text.

- **Unigrams (1-grams)**: Single words. Example: "The cat sat on the mat" → "The", "cat", "sat", "on", "the", "mat".
- **Bigrams (2-grams)**: Pairs of words. Example: "The cat", "cat sat", "sat on", "on the", "the mat".
- **Trigrams (3-grams)**: Triplets of words. Example: "The cat sat", "cat sat on", "sat on the", "on the mat".

Usage: n-grams are used to model the likelihood of sequences and predict the next word in a sequence based on the previous words.

Basics of Grammar-Free Analyzers

2. Term Frequency-Inverse Document Frequency (TF-IDF)

Concept: Measures the importance of a word in a document relative to a collection of documents (corpus).

- **Term Frequency (TF)**: The frequency of a term in a document.
- **Inverse Document Frequency (IDF)**: Measures how much information the word provides, i.e., whether the term is common or rare across all documents.

Formula:

$$\text{TF-IDF} = \text{TF} \times \text{IDF}$$

$$\text{IDF} = \log\left(\frac{N}{n}\right)$$

Where NNN is the total number of documents and nnn is the number of documents containing the term.

Usage: TF-IDF is used for text mining and information retrieval to identify important words in documents.

Basics of Grammar-Free Analyzers

3. Word Embeddings

Concept: Represents words as dense vectors of real numbers in a continuous vector space, capturing semantic meanings.

- **Word2Vec**: Creates word embeddings using two models, CBOW (Continuous Bag of Words) and Skip-gram. It captures the context of words.
- **GloVe (Global Vectors for Word Representation)**: Constructs word embeddings by aggregating global word-word co-occurrence statistics from a corpus.
- **FastText**: Extends Word2Vec by considering subword information, making it effective for morphologically rich languages.

Usage: Word embeddings are used in various NLP tasks such as text classification, sentiment analysis, and machine translation.

Basics of Grammar-Free Analyzers

YENEPOYA
(DEEMED TO BE UNIVERSITY)

4. Latent Dirichlet Allocation (LDA)

Concept: A generative probabilistic model used for topic modeling. It assumes documents are mixtures of topics and topics are mixtures of words.

Process:

- **Step 1**: Determine the number of topics.
- **Step 2**: Assign words to topics based on their distribution.
- **Step 3**: Iterate to improve the assignment.

Usage: LDA is used to identify underlying topics in large text corpora, making it useful for organizing, summarizing, and understanding large sets of documents.

Basics of Grammar-Free Analyzers

YENEPOYA
(DEEMED TO BE UNIVERSITY)

5. Hidden Markov Models (HMMs)

Concept: A statistical model that represents systems with hidden states through observable events. In NLP, it's used for tasks like part-of-speech tagging.

Components:

- **States**: Represent parts of speech or other categories.
- **Observations**: Actual words or tokens in the text.
- **Transitions**: Probabilities of moving from one state to another.
- **Emissions**: Probabilities of an observation being produced from a state.

Usage: HMMs are applied to sequence labeling tasks such as part-of-speech tagging, named entity recognition, and bioinformatics.

Basics of Grammar-Free Analyzers

6. Naive Bayes Classifier

Concept: A probabilistic classifier based on Bayes' theorem, assuming independence between features.

Formula: $P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$

Where $P(C|X)$ is the probability of class C given feature set X.

Usage: Naive Bayes is used for text classification tasks such as spam detection, sentiment analysis, and document categorization.

Basics of Grammar-Free Analyzers

7. Pointwise Mutual Information (PMI)

Concept: Measures the association between two words by comparing the probability of their co-occurrence with their individual probabilities.

Formula:

$$PMI(x, y) = \log\left(\frac{P(x,y)}{P(x) \cdot P(y)}\right)$$

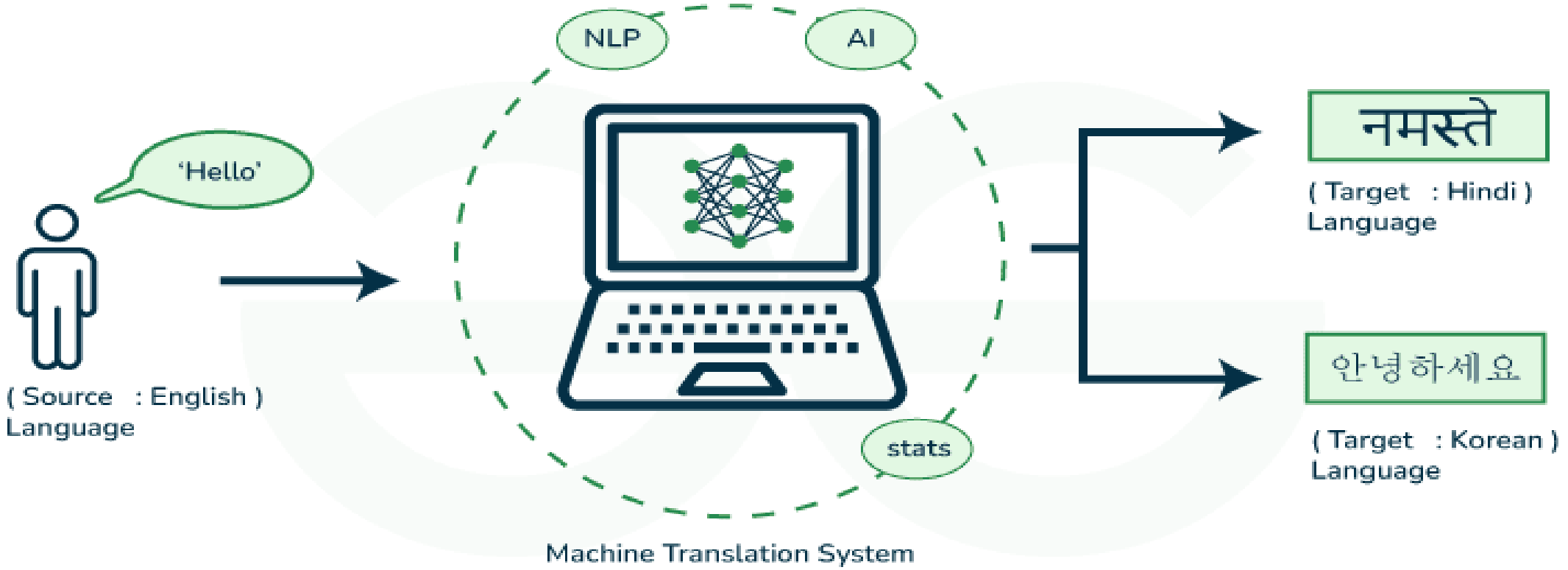
Usage: PMI is used to find collocations and word associations, helping in tasks like word sense disambiguation and information retrieval.

Basics of translation

What is Machine Translation?

Machine translation is a sub-field of computational linguistics that focuses on developing systems capable of automatically translating text or speech from one language to another. In Natural Language Processing (NLP), the goal of machine translation is to produce translations that are not only grammatically correct but also convey the meaning of the original content accurately.

Basics of translation



Basics of translation

What are the key approaches in Machine Translation?

In machine translation, the original text is decoded and then encoded into the target language through two step process that involves various approaches employed by language translation technology to facilitate the translation mechanism.

1. Rule-Based Machine Translation

Rule-based machine translation relies on these resources to ensure precise translation of specific content. The process involves the software parsing input text, generating a transitional representation, and then converting it into the target language with reference to grammar rules and dictionaries.

Basics of translation

2. Statistical Machine Translation

- Rather than depending on linguistic rules, statistical machine translation utilizes machine learning for text translation. Machine learning algorithms examine extensive human translations, identifying statistical patterns.
- When tasked with translating a new source text, the software intelligently guesses based on the statistical likelihood of specific words or phrases being associated with others in the target language.

Basics of translation

3. Neural Machine Translation (NMT)

- A neural network, inspired by the human brain, is a network of interconnected nodes functioning as an information system. Input data passes through these nodes to produce an output.
- Neural machine translation software utilizes neural networks to process vast datasets, with each node contributing a specific change from source text to target text until the final result is obtained at the output node.

Basics of translation

4. Hybrid Machine Translation

- Hybrid machine translation tools integrate multiple machine translation models within a single software application, leveraging a combination of approaches to enhance the overall effectiveness of a singular translation model.
- This process typically involves the incorporation of rule-based and statistical machine translation subsystems, with the ultimate translation output being a synthesis of the results generated by each subsystem.

Basics of translation

Why we need Machine Translation in NLP?

Machine translation in Natural Language Processing (NLP) has several benefits, including:

1. **Improved communication:** Machine translation makes it easier for people who speak different languages to communicate with each other, breaking down language barriers and facilitating international cooperation.
2. **Cost savings:** Machine translation is typically faster and less expensive than human translation, making it a cost-effective solution for businesses and organizations that need to translate large amounts of text.

Basics of translation

3. Increased accessibility: Machine translation can make digital content more accessible to users who speak different languages, improving the user experience and expanding the reach of digital products and services.

4. Improved efficiency: Machine translation can streamline the translation process, allowing businesses and organizations to quickly translate large amounts of text and improving overall efficiency.

5. Language learning: Machine translation can be a valuable tool for language learners, helping them to understand the meaning of unfamiliar words and phrases and improving their language skills.

References:

- E. Rich and K. Knight, “Artificial intelligence”, TMH, 2nd ed., 1992.
- Nilsson, N. J. (1986). Principles of artificial intelligence. Morgan Kaufmann.
- Craig, J. J. (2009). Introduction to robotics: mechanics and control, 3/E. Pearson Education India.
- Klafter, R. D., Chmielewski, T. A., &Negin, M. (1989). Robotic engineering : an integrated approach. Prentice-Hall.
- Yoshikawa, T. (1990). Foundations of robotics: analysis and control. MIT press.