Unit V: Applications of AI (8 Hours)

5.1 Expert Systems, Components of Expert System, Steps in Development of Expert Systems

5.2.Natural Language Processing:
5.2.1 Natural Language Understanding and Natural Language Generation
5.2.2 Steps of Natural Language Processing: Lexical Analysis (Segmentation, Morphological Analysis), Syntactic Analysis, Semantic Analysis, Pragmatic Analysis

5.3 Machine Vision Concepts:
5.3.1 Machine vision and its applications,
5.3.2 Components of Machine Vision System

5.4 Robotics: Robot Hardware (Sensors and Effectors) , Robotic Perceptions

5.1 Expert Systems, Components of Expert System, Steps in Development of Expert Systems

Expert systems are a crucial subset of artificial intelligence (AI) that simulate the decision-making ability of a human expert.

These systems use a knowledge base filled with domain-specific information and rules to interpret and solve complex problems.

For example, a medical expert system can analyze a patient's symptoms and suggest possible diagnoses or treatments.

Similarly, a financial expert system can evaluate market trends and recommend investment strategies.

Components and Architecture of Expert System

An expert system is made up of several interconnected components, each playing a crucial role in its functionality.

1. Knowledge Base: The Heart of the System

The knowledge base is the heart of an expert system.

It contains all the facts, rules, and expert knowledge related to a specific domain.

Think of it as a library filled with textbooks, research papers, and expert opinions. The accuracy and completeness of the knowledge base directly impact the system's performance.

If the knowledge is outdated or incomplete, the system's recommendations may be flawed.

In a financial expert system, the knowledge base might include rules for detecting fraudulent transactions, such as "If a transaction exceeds $10,000 and occurs in a foreign country, flag it for review."

2. Inference Engine: The Brain Behind the Decisions

The inference engine is the brain of the expert system.

It processes the information stored in the knowledge base to draw conclusions or make recommendations.
The inference engine uses reasoning strategies (like forward chaining or backward chaining) to analyze data and apply rules.

Forward Chaining:
Starts with available data and works toward a conclusion.
For example, "If the temperature is high and the patient has a cough, diagnose a respiratory infection."
Backward Chaining:
Starts with a goal and works backward to find supporting evidence.
For example, "If the goal is to diagnose diabetes, check for symptoms like frequent urination and high blood sugar."

3. User Interface: Bridging the Gap Between System and User

The user interface is the bridge that allows users to interact with the expert system.
It’s designed to be intuitive and user-friendly, ensuring that even non-experts can use the system effectively.
Users provide a query (problem or question), and the system processes the request.
The system then delivers advice or recommendations back to the user.

4. Explanation Module: Building Trust Through Transparency

The explanation module is a critical feature that explains how the system arrived at a particular conclusion.
It’s like a teacher showing their work when solving a math problem.
This module provides users with a clear, step-by-step explanation of the system’s reasoning.

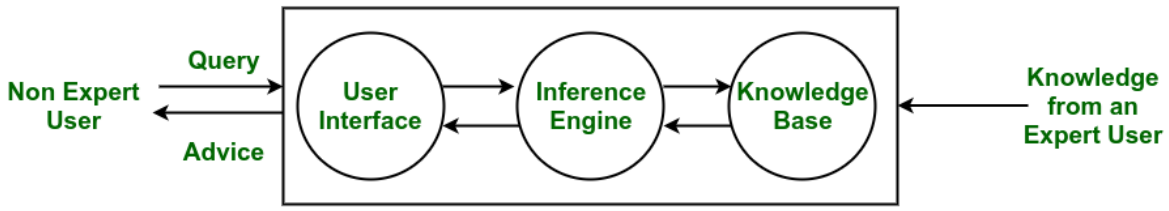
This transparency is especially important in fields like healthcare and finance, where decisions can have significant consequences.

Example: A medical expert system might explain, "I diagnosed pneumonia because the patient has a fever, cough, and abnormal chest X-ray."

5. Knowledge Acquisition Module: Keeping the System Up-to-Date

The knowledge acquisition module is responsible for updating and expanding the knowledge base.
It ensures that the system stays current with the latest information and trends. Without regular updates, the system's knowledge base can become outdated, reducing its effectiveness.

Let's understand it's architecture with help of diagram:



Expert Systems in AI : Architecture

The working mechanism of an expert system begins when a non-expert user submits a query through the user interface.

- This query is then processed by the inference engine, which applies logical rules and reasoning techniques to analyze the input.
- The inference engine interacts with the knowledge base, retrieving relevant facts, rules, and heuristics contributed by expert users.
- Based on this structured knowledge, the system derives conclusions and formulates an appropriate response.

Finally, the expert system provides advice or recommendations to the user, assisting in decision-making or problem-solving without requiring direct human expert intervention.

Steps in Development of Expert Systems

The following points highlight the five main stages to develop an expert system.The stages are: 1. Identification 2. Conceptualisation 3. Formalisation (Designing) 4. Implementation 5. Testing (Validation, Verification and Maintenance).

The knowledge engineer and the domain expert usually work very closely together for long periods of time throughout the several stages of the development process.
An expert system is developed and refined over a period of several years since it is typically a computer-based software.

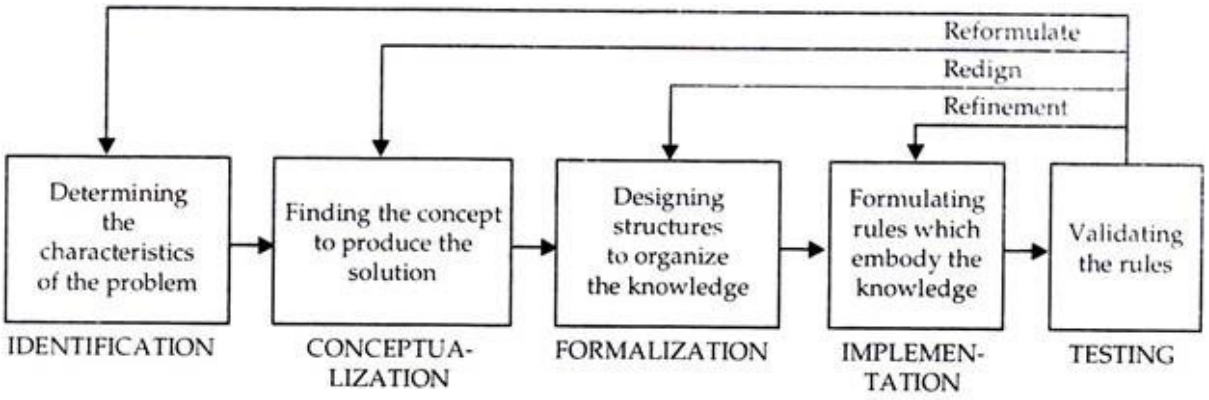


Fig. 12.8. *The five stages of expert system development.*

1. Identification:
Before we can begin to develop an expert system, it is important to describe, with as much precision as possible, the problem which the system is intended to solve. It is not enough simply to feel that an expert system would be helpful in a certain situation; we must determine the exact nature of the problem and state the precise goals which indicate exactly how the expert system is expected to contribute to the solution.
To begin, the knowledge engineer, who may be unfamiliar with this particular domain, consults manuals and training guides to gain some familiarity with the subject. Then the domain expert describes several typical problem states. The knowledge engineer attempts to extract fundamental concepts from the similar cases in order to develop a more general idea of the purpose of the expert system.

After the domain expert describes several cases, the knowledge engineer develops a 'first-pass' problem description. Typically, the domain expert may feel that the description does not entirely represent the problem. The domain expert then suggests changes to the description and provides the knowledge engineer with additional examples to illustrate further the problem's fine points

Next, the knowledge engineer revises the description, and the domain expert suggests further changes. This process is repeated until the domain expert is satisfied that the knowledge engineer understands the problems and until both are satisfied that the description adequately portrays the problem which the expert system is expected to solve.

This 'iterative' procedure (Fig. 12.9) is typical of the entire expert-system development process. The results are evaluated at each stage of the process and compared to the expectations. If the results do not meet the expectations, adjustments are made to that stage of the process, and the new results are evaluated. The process continues until satisfactory results are achieved.

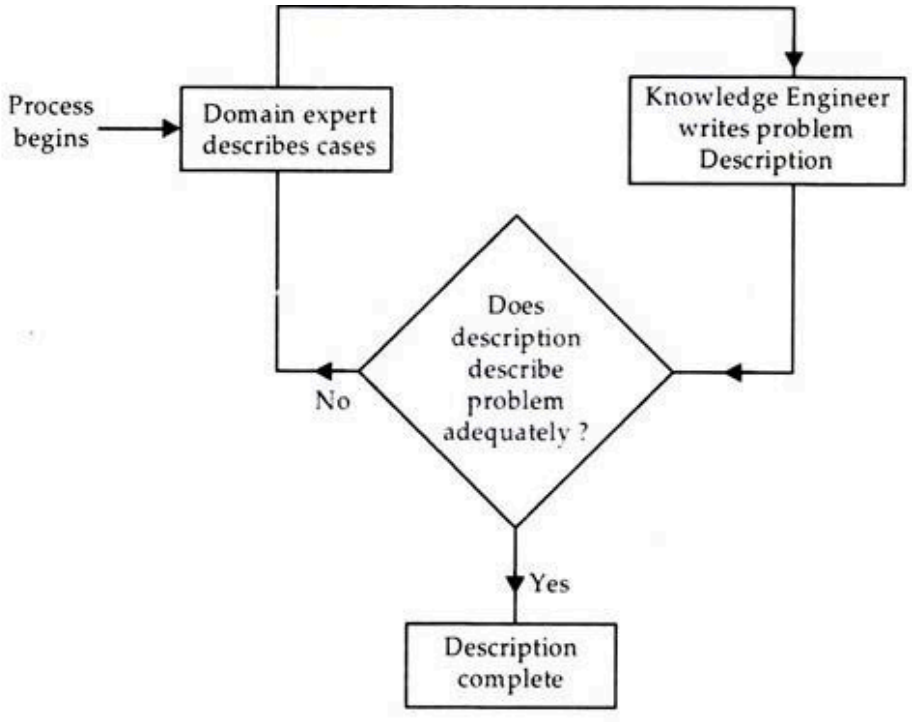


Fig. 12.9. *The iterative process of identifying the problem which the expert system is to solve.*

2. Conceptualisation:
Once it has been identified for the problem an expert system is to solve, the next stage involves analysing the problem further to ensure that its specifics, as well as generalities, are understood.

In the conceptualisation stage, the knowledge engineer frequently creates a diagram of the problem to depict graphically the relationships between the objects and processes in the problem domain. It is often helpful at this stage to divide the problem into a series of sub-problems and to diagram both the relationships among the pieces of each sub-problem and the relationships among the various sub-problems.

As in the identification stage, the conceptualisation stage involves a circular procedure of iteration and reiteration between the knowledge engineer and the domain expert. When both agree that the key concepts-and the relationships among them-have been adequately conceptualised, this stage is complete.

Not only is each stage in the expert system development process circular, the relationships among the stages may be circular as well. Since each stage of the development process adds a level of detail to the previous stage, any stage may expose a weakness in a previous stage.

For example, a problem with the description generated in the identification stage may be discovered during conceptualisation. A key element of the description may have been omitted, or perhaps a goal was stated incorrectly. If this occurs, a brief return to the identification stage is required to increase the accuracy of the description (Fig. 12.10). A similar process can occur in any stage of development.

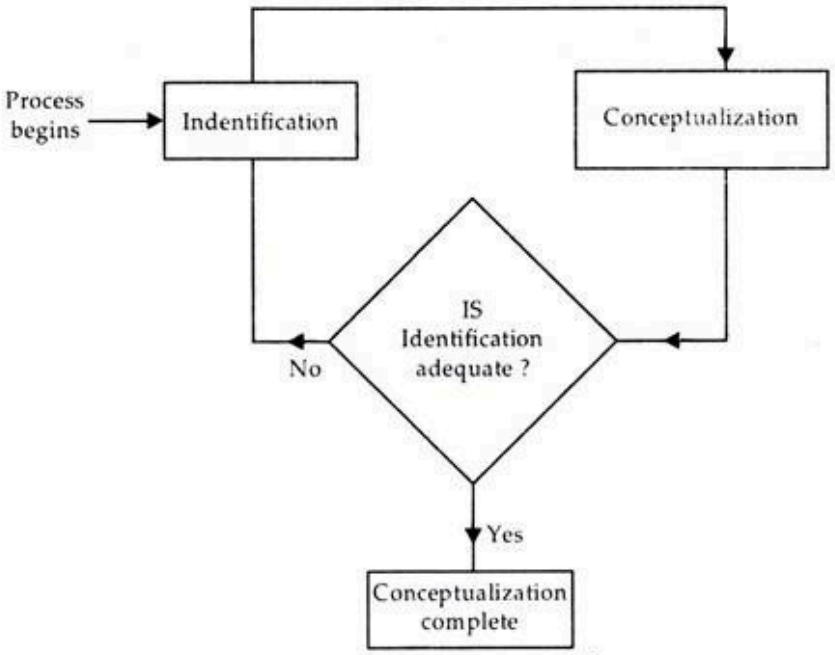


Fig. 12.10. *The iterative relationship between the indentification and conceptualisation stages of expert system development.*

3. Formalisation (Designing):
In the preceding stages, no effort has been made to relate the domain problem to the artificial intelligence technology which may solve it. During the identification and formalization stages, the focus is entirely on understanding the problem. Now, during the formalization stage, the problem is connected to its proposed solution, an expert system is supplied by analyzing the relationships depicted in the conceptualization stage. The knowledge engineer begins to select the techniques which are appropriate for developing this particular expert system.

During formalization, it is important that the knowledge engineer be familiar with the following:

1. The various techniques of knowledge representation and intelligent search techniques used in expert systems.

2. The expert system tools which can greatly expedite the development process.

3. Other expert systems which may solve similar problems and thus may be adaptable to problem at hand.

Often it is desirable to select a single development technique or tool which can be used throughout all segments of the expert system. However, the knowledge engineer may determine that no particular technique is appropriate for the entire expert system, making it necessary to use different techniques for different sub-problems. Once it has been determined which technique(s) will be used the knowledge engineer starts to develop a formal specification which can used to develop a prototype expert system.

In the case of a rule-based system, for example, the knowledge engineer develops a set of rules designed to represent the knowledge communicated by the domain expert. This is a critical part of the development process, requiring, great skill on the part-of the knowledge engineer. Many domain experts can explain what they do but not why; therefore, one of the knowledge engineer's primary responsibilities is to analyse example situations and filter in from those examples a set of rules which describe the domain expert's knowledge.

The formalisation process is often the most interactive stage of expert system development, as well as the most time consuming. The knowledge engineer must develop a set of rules and ask the domain expert if those rules adequately represent the expert's knowledge. The domain expert reviews the rules proposed by the knowledge engineer and suggests changes, which are then incorporated into the knowledge base by the knowledge engineer.

As in the other development stages, this process also is iterative: the rule review is repeated and the rules are refined continually until the results are satisfactory. It is not unusual for the formalisation process of a complex expert system to last for several years. (Fig. 12.10).

4. Implementation:
During the implementation stage the formalised concepts are programmed into the computer which has been chosen for system development, using the predetermined techniques and tools to implement a 'first-pass' (prototype) of the expert system.

Theoretically, if the methods of the previous stages have been followed with diligence and care, the implementation of the prototype should proceed

smoothly. In practice, the development of an expert system may be as much an art as it is a science, because following all the rules does not guarantee that the system will work the first time it is implemented. In fact, experience suggests the opposite. Many scientists actually consider the prototype to be a 'throw-away' system, useful for evaluating progress but hardly a usable expert system.

If the prototype works at all, the knowledge engineer may be able to determine if the techniques chosen to implement the expert system were the appropriate ones. On the other hand, the knowledge engineer may discover that the chosen techniques simply cannot be implemented. It may not be possible, for example, to integrate the knowledge representation techniques selected for different sub-problems. At that point, the concepts may have to be re-formalised, or it even may be necessary to create new development tools to implement the system efficiently. The implementation stage is illustrated in Fig. 12.11.

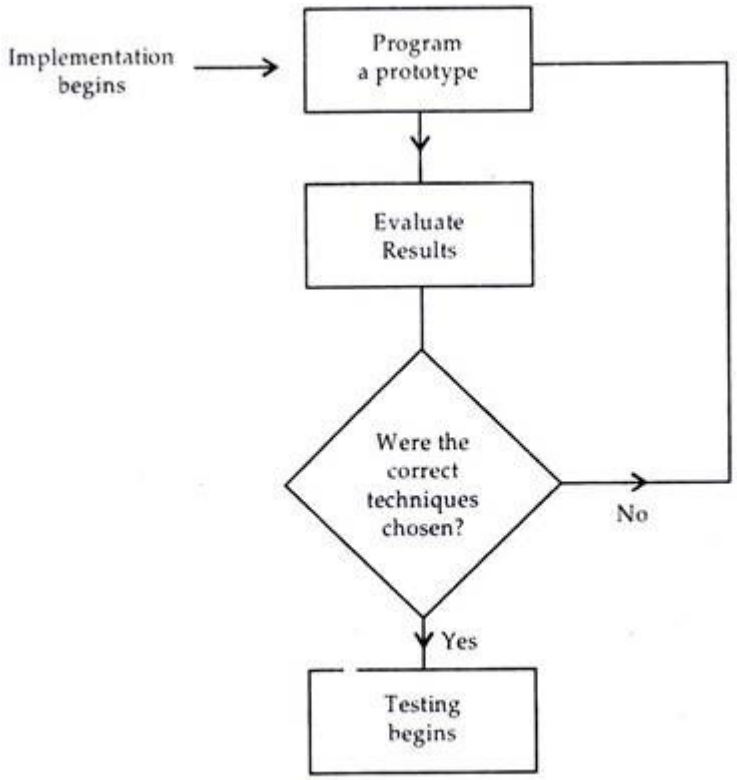


Fig. 12.11. *The implementation stage of expert system development.*

Once the prototype system has been refined sufficiently to allow it to be executed, the expert system is ready to be tested thoroughly to ensure that it expertise's correctly.

5. Testing (Validation, Verification and Maintenance):
The chance of prototype expert system executing flawlessly the first time it is tested are so slim as to be virtually non-existent. A knowledge engineer does not expect the testing process to verify that the system has been constructed entirely correctly. Rather, testing provides an opportunity to identify the weaknesses in the structure and implementation of the system and to make the appropriate corrections.

Depending on the types of problems encountered, the testing procedure may indicate that the system was implemented incorrectly, or perhaps that the rules were implemented correctly but were poorly or incompletely formulated. Results from the tests are used as 'feedback' to return to a previous stage and adjust the performance of the system.

Once the system has proven to be capable of correctly solving straight-forward problems, the domain expert suggests complex problems which typically would require a great deal of human expertise. These more demanding tests should uncover more serious flaws and provide ample opportunity to 'fine tune' the system even further.

Ultimately, an expert system is judged to be entirely successful only when it operates at the level of a human expert. The testing process is not complete until it indicates that the solutions suggested by the expert system are consistently as valid as those provided by a human domain expert.

5.2.Natural Language Processing:

Natural Language Processing (NLP) is a subfield of computer science and artificial intelligence (AI) that focuses on the interaction between computers and human language.
It enables machines to understand, interpret, and generate human language in a way that is both meaningful and useful.

Examples:
Chatbots
Voice assistants (Siri, Alexa)

5.2.1 Natural Language Understanding and Natural Language Generation

Natural language understanding is a subset of natural language processing, which uses syntactic and semantic analysis of text and speech to determine the meaning of a sentence. Syntax refers to the grammatical structure of a sentence, while semantics alludes to its intended meaning. NLU also establishes a relevant ontology: a data structure which specifies the relationships between words and phrases. While humans naturally do this in conversation, the combination of these analyses is required for a machine to understand the intended meaning of different texts.

Our ability to distinguish between homonyms and homophones illustrates the nuances of language well. For example, let's take the following two sentences:

Alice is swimming against the current.
The current version of the report is in the folder.
In the first sentence, the word, current is a noun. The verb that precedes it, swimming, provides additional context to the reader, allowing us to conclude that we are referring to the flow of water in the ocean. The second sentence uses the word current, but as an adjective. The noun it describes, version, denotes multiple iterations of a report, enabling us to determine that we are referring to the most up-to-date status of a file.

These approaches are also commonly used in data mining to understand consumer attitudes. In particular, sentiment analysis enables brands to monitor their customer feedback more closely, allowing them to cluster positive and negative social media comments and track net promoter scores. By reviewing comments with negative sentiment, companies are able to identify and address potential problem areas within their products or services more quickly.

Natural language generation is another subset of natural language processing. While natural language understanding focuses on computer reading comprehension, natural language generation enables computers to write. NLG is the process of producing a human language text response based on some data input. This text can also be converted into a speech format through text-to-speech services.

NLG also encompasses text summarization capabilities that generate summaries from in-put documents while maintaining the integrity of the information. Extractive summarization is the AI innovation powering Key Point Analysis used in That's Debatable.

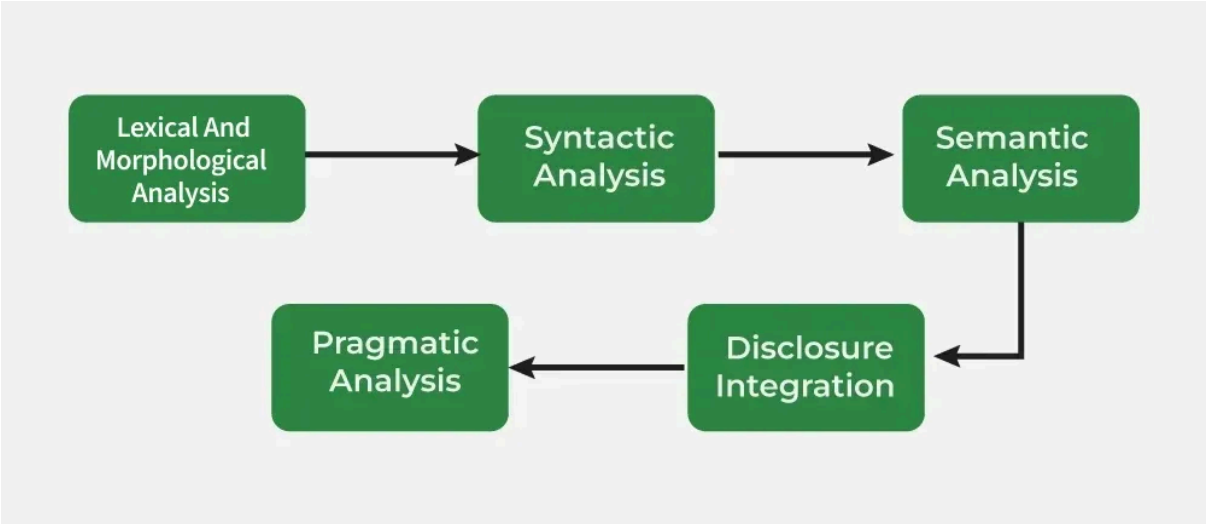
Initially, NLG systems used templates to generate text. Based on some data or query, an NLG system would fill in the blank, like a game of Mad Libs. But over time, natural language generation systems have evolved with the application of hidden Markov chains, recurrent neural networks, and transformers, enabling more dynamic text generation in real time.

As with NLU, NLG applications need to consider language rules based on morphology, lexicons, syntax and semantics to make choices on how to phrase responses appropriately. They tackle this in three stages:

- Text planning: During this stage, general content is formulated and ordered in a logical manner.
- Sentence planning: This stage considers punctuation and text flow, breaking out content into paragraphs and sentences and incorporating pronouns or conjunctions where appropriate.
- Realization: This stage accounts for grammatical accuracy, ensuring that rules around punctation and conjugations are followed. For example, the past tense of the verb run is ran, not runned.

5.2.2 Steps of Natural Language Processing: Lexical Analysis (Segmentation, Morphological Analysis), Syntactic Analysis, Semantic Analysis, Pragmatic Analysis

Natural Language Processing (NLP) helps computers to understand, analyze and interact with human language. It involves a series of phases that work together to process language and each phase helps in understanding structure and meaning of human language. In this article, we will understand these phases.



steps-of-NLP

1. Lexical and Morphological Analysis

Lexical Analysis
It focuses on identifying and processing words (or lexemes) in a text.
It breaks down the input text into individual tokens that are meaningful units of language such as words or phrases.

Key tasks in Lexical analysis:

1. Tokenization: Process of dividing a text into smaller chunks called tokens. For example the sentence "I love programming" would be tokenized into ["I", "love", "programming"].
2. Part-of-Speech Tagging: Assigning parts of speech such as noun, verb, adjective to each token in the sentence. This helps us to understand grammatical roles of words in the context.

Example: Consider the sentence: "I am reading a book."

- Tokenization: Sentence is broken down into individual tokens or words: ["I", "am", "reading", "a", "book"]
- Part-of-Speech Tagging: Each token is assigned a part of speech: ["I" → Pronoun (PRP), "am" → Verb (VBP), "reading" → Verb (VBG), "a" → Article (DT), "book" → Noun (NN)]

Importance of Lexical Analysis

- Word Identification: It breaks text into tokens which helps the system to understand individual words for further processing.
- Text Simplification: It simplifies text through tokenization and stemming which improves accuracy in NLP tasks.

Morphological Analysis
It deals with morphemes which are the smallest units of meaning in a word.
It is important for understanding the structure of words and their parts by identifying free morphemes (independent words like "cat") and bound morphemes (like prefixes or suffixes e.g. "un-" or "-ing").

Key tasks in morphological analysis:

1. Stemming: Reducing words to their root form like "running" to "run".
2. Lemmatization: Converting words to their base or dictionary form considering the context like "better" becomes "good".

Importance of Morphological Analysis

- Understanding Word Structure: It helps in breaking the composition of complex words.
- Improving Accuracy: It enhances accuracy of tasks such as part-of-speech tagging, syntactic parsing and machine translation.

By identifying and analyzing morphemes system can identify text correctly at the most basic level which helps in more advanced NLP applications.

2. Syntactic Analysis (Parsing)

Syntactic Analysis helps in understanding how words in a sentence are arranged according to grammar rules.
It ensures that the sentence follows correct grammar which makes the meaning clearer.

The goal is to create a parse tree which is a diagram showing the structure of sentence.
It breaks the sentence into parts like the subject, verb and object and shows how these parts are connected.
This helps machines understand the relationships between words in the sentence.

Key components of syntactic analysis include:

1. POS Tagging: Assigning parts of speech (noun, verb, adjective) to words in a sentence as discussed earlier.
2. Ambiguity Resolution: Handling words that have multiple meanings (e.g "book" can be a noun or a verb).

Examples
Consider the following sentences:

Correct Syntax: "John eats an apple."
Incorrect Syntax: "Apple eats John an."

Despite using same words only the first sentence is grammatically correct and makes sense. The correct arrangement of words according to grammatical rules is what makes the sentence meaningful. By analyzing sentence structure NLP systems can better understand and generate human language. This helps in tasks like machine translation, sentiment analysis and information retrieval by making the text clearer and reducing confusion.

3. Semantic Analysis

Semantic Analysis focuses on understanding meaning behind words and sentences. It ensures that the text is not only grammatically correct but also logically coherent and contextually relevant. It aims to understand dictionary definitions of words and their usage in context and also find whether the arrangement of words in a sentence makes logical sense.

Key Tasks in Semantic Analysis

1. Named Entity Recognition (NER): It identifies and classifies entities such as names of people, locations, organizations, dates and more. These entities provide important meaning in the text and help in understanding the context. For example in the sentence "Tesla announced its new electric vehicle in California," NER would identify "Tesla" as an organization and "California" as a location.
2. Word Sense Disambiguation (WSD): Many words have multiple meanings depending on the context in which they are used. It identifies the correct meaning of a word based on its surrounding text. For example word "bank" can refer to a financial institution or the side of a river. It uses context to identify which meaning applies in a given sentence which ensures that interpretation is accurate.

Example of Semantic Analysis

"Apple eats a John." while grammatically correct this sentence doesn't make sense semantically because an apple cannot "eat" a person. Semantic analysis ensures that the meaning is logically sound and contextually appropriate. It is important for various NLP applications including machine translation, information retrieval and question answering.

4. Discourse Integration

It is the process of understanding how individual sentences or segments of text connect and relate to each other within a broader context.
This phase ensures that the meaning of a text is consistent and coherent across multiple sentences or paragraphs.
It is important for understanding long or complex texts where meaning focuses on previous statements.

Key aspects of discourse integration:

1. Anaphora Resolution: Anaphora refers to the use of pronouns or other references that depend on earlier parts of the text. For example in the sentence "Taylor went to the store. She bought groceries" pronoun "She" refers back to "Taylor." It ensures that references like these are correctly understood by linking them to their antecedents.
2. Contextual References: Many words or phrases can only be fully understood when considered in the context of following sentences. It helps in interpreting how certain words or phrases focuses on context. For example "It was a great day" is clearer when you know what event or situation is being discussed.

Example of Discourse Integration

1. "Taylor went to the store to buy some groceries. She realized she forgot her wallet." Understanding that "Taylor" is the antecedent of "she" is important for understanding sentence's meaning.
2. "This is unfair!" helps in understand what "this" refers to we need to identify following sentences. Without context statement's meaning remains unclear.

It is important for NLP applications like machine translation, chatbots and text summarization. It ensures that meaning remains same across sentences which helps machines to understand context. This enables accurate and natural responses in applications like conversational AI and document translation.

5. Pragmatic Analysis

Pragmatic analysis helps in understanding the deeper meaning behind words and sentences by looking beyond their literal meanings. While semantic analysis looks at the direct meaning it considers the speaker's or writer's intentions, tone and context of the communication.

Key tasks in pragmatic analysis:

- Understanding Intentions: Sometimes language doesn’t mean what it says literally. For example when someone asks "Can you pass the salt?" it's not about ability but a polite request. It helps to understand true intention behind such expressions.
- Figurative Meaning: Language often uses idioms or metaphors that can’t be taken literally.

Examples of Pragmatic Analysis

"Hello! What time is it?" here it might be a straightforward request for the current time but it could also imply concern about being late.
For example "I’m falling for you" means "I love you" not literally falling. It helps to interpret these non-literal meanings.

5.3 Machine Vision Concepts:

Machine vision, also known as computer vision, is a field of artificial intelligence that enables machines to interpret and understand visual information.
It involves the analysis, modification, and understanding of images to control systems or provide more informative images.
The field has vast applications, including automotive systems, photography, video surveillance, biometrics, and more.
Machine vision systems are designed to recognize objects from acquired image data and perform useful tasks based on that recognition.

5.3.1 Machine vision and its applications

Machine vision refers to the technology and methods used to enable machines to interpret and understand visual information from the world, mimicking human vision capabilities.
It involves the use of hardware and software systems that capture, process, and analyze images to make decisions based on visual data. This technology is crucial in automation, quality control, and various industrial processes.

its applications

- Manufacturing: The Quality control assembly verification and defect detection are crucial aspects of the manufacturing processes. The Machine vision systems can be employed to the inspect products for the defects verify proper assembly and ensure overall quality throughout the production line.

- Automotive: The Automated inspection of the vehicle components using the machine vision technology helps ensure the quality and reliability of the automotive parts. This includes the inspecting parts for the defects, verifying dimensions and detecting any abnormalities during the manufacturing process.

- Healthcare: In healthcare, machine vision plays a vital role in the medical imaging and diagnostics. It enables the analysis of the medical images such as the X-rays, MRIs and CT scans to the aid in diagnosing diseases or detecting abnormalities and guiding medical procedures with the precision.

- Agriculture: The Machine vision systems are used in the agriculture for the crop monitoring and sorting. These systems can analyze images of the crops to the assess their health or diseases and optimize harvesting processes.

5.3.2 Components of Machine Vision System

Machine vision systems consist of several crucial components:

- Camera: The camera is the primary input device in the machine vision system. It captures images of objects or scenes to be inspected. These cameras come in various types including area scan cameras and line scan cameras depending on the specific application.

- Illumination: The Proper lighting is critical to ensure that the camera captures high-quality images. Different illumination methods such as

LED lights and laser lights or halogen lights can be used to the highlight the object of the interest effectively.

- Lenses: The Lenses are used to focus and control the image formation. They determine the field of view and depth of field in which can be adjusted based on specific inspection requirements.

- Vision Processors: These are high-speed computers or processors that handle image acquisition, preprocessing and analysis. They run the algorithms responsible for the detecting defects or anomalies in images.

- Frame Grabbers: The Frame grabbers are used to capture and digitize analog camera outputs making them suitable for the image processing by the vision processor.

- Software: The Specialized software is employed to process and analyze the images, detect defects and make decisions. This software often includes image recognition, pattern matching and machine learning capabilities.

5.4 Robotics: Robot Hardware (Sensors and Effectors) , Robotic Perceptions

Robotics is an interdisciplinary engineering branch that deals with the design, construction, working, and use of robots. Being an interdisciplinary branch, it combines several engineering disciplines such as mechanical engineering, electrical engineering, control engineering, electronics engineering, computer science, communication engineering, material engineering, etc.
In robotics, mechanical engineering contributes in design and construction of physical structure of robots, electrical and electronics engineering provides interconnection between different parts, while computer engineering provides the algorithms and programs for automation and operation of the robots.

Robotic hardware consists of sensors that detect environmental information and effectors (actuators) that allow robots to act, forming the foundation of robotic perception and interaction.

Robot Hardware: Sensors

Sensors are devices that collect information about the robot's internal state or external environment, enabling feedback and intelligent decision-making. They are broadly categorized as follows:

- Proprioceptive Sensors: Measure the robot's own state, such as joint angles, position, velocity, torque, or battery levels. Examples include encoders, gyroscopes, and accelerometers. These sensors are critical for balancing, navigation, and coordinated motion.

- Exteroceptive Sensors: Acquire data about the surrounding environment. Common types include:
    - Vision Sensors (Cameras): Capture images or video streams to detect objects, recognize patterns, or navigate using computer vision techniques.

    - Distance/Range Sensors: Include LiDAR, ultrasound, and infrared sensors to measure distances to nearby objects, useful for obstacle avoidance and mapping.

    - Tactile Sensors: Detect contact, pressure, or force, often incorporated into robot hands or grippers for manipulation tasks.

    - Environmental Sensors: Measure temperature, humidity, gas concentrations, or light, enabling context-aware reactions.

Robot Hardware: Effectors (Actuators)

Effectors, or actuators, convert electrical or control signals from the robot's controller into physical movement or actions. They are categorized according to the type of motion they produce:

- Electric Motors: Provide rotational or linear motion for wheels, arms, or joints. Common types are DC motors, stepper motors, and servo motors, widely used for precision and controlled movement.

- Hydraulic Actuators: Use pressurized fluid for high-force motions, suitable for heavy-duty industrial robots.

- Pneumatic Actuators: Use compressed air, often for lightweight, fast, and simple movements.

- Shape Memory Alloys and Piezoelectric Actuators: Provide micro- or nanoscale movements for specialized tasks.

- Effectors commonly include robotic arms, grippers, mobile platforms, or end-effectors designed to interact with specific objects or environments.

Robotic Perception

Robotic perception is the process of interpreting sensor data to understand the environment and make decisions. It integrates hardware and software to enable tasks such as navigation, manipulation, and autonomous interaction. Key aspects include:

- Sensor Fusion: Combining data from multiple sensors (e.g., LiDAR and camera) for accurate perception and reducing uncertainty.

- Localization and Mapping: Robots use sensors to determine their position (localization) and construct a map of the environment (SLAM – Simultaneous Localization and Mapping).

- Object Recognition and Tracking: Vision and tactile sensors allow robots to identify and track objects for manipulation or interaction.

- Environmental Awareness and Decision-Making: Sensor data informs algorithms that enable obstacle avoidance, motion planning, and adaptive behaviors.

Robotic perception bridges the gap between raw sensor readings and actionable insights, allowing robots to respond intelligently to dynamic environments.