Applied AI Unit 1

Applied AI Unit 1

Applied Artificial Intelligence

Unit 1

Artificial Intelligence (AI)

- Branch of computer science that focuses on creating systems capable of performing tasks that typically require human intelligence.
- Tasks include learning, reasoning, problem-solving, perception, language understanding, and decision-making.
- Use algorithms and models to analyze data, recognize patterns, and make autonomous decisions.

AI is classified into different types based on its capabilities and functionalities.

Based on Capabilities

(i) Narrow AI (Weak AI)

- Designed for a specific task and cannot perform beyond its programming.
- Ex.: Voice assistants like Siri, Alexa, and Google Assistant (process speech but don't have general intelligence.

(ii) General AI (Strong AI)

- Aims to replicate human intelligence and can perform any intellectual task that a human can.
- Example: A future AI that can think, reason, and make independent decisions in different domains.

(iii) Super AI

- A hypothetical AI that surpasses human intelligence in all aspects, including creativity, emotional intelligence, and decision-making.
- Ex.: AI surpassing humans in research, innovation, and governance, often seen in science fiction.

Types of AI

2. Based on Functionalities

(i) Reactive Machines

- Can respond to specific situations but do not have memory or learning ability.
- Example: IBM's Deep Blue, which defeated chess champion Garry Kasparov in 1997.

(ii) Limited Memory

- Can learn from past experiences and make better decisions over time.
- Example: Self-driving cars, which use past data to recognize road signs and obstacles.

(iii) Theory of Mind (Future AI)

- AI that understands human emotions, beliefs, and social interactions.
- Example: AI robots that can form relationships with humans, like Sophia (Hanson Robotics).

(iv) Self-Aware AI

- AI with self-consciousness and emotions (not yet achieved).
- Example: A robot developing its own goals and self-preservation mechanisms.

How Does AI Work?

AI works using a combination of algorithms, machine learning, and data processing.

1. Machine Learning (ML)

- AI learns from past data and improves over time without explicit programming.
- Example: Netflix's recommendation system, which suggests shows based on viewing history.

2. Deep Learning

- A subset of ML that uses artificial neural networks for complex problem-solving.
- Example: Facial recognition in smartphones, which identifies users using deep learning.

3. Natural Language Processing (NLP)

- Helps AI understand and generate human language.
- Example: ChatGPT, Google Translate, and Grammarly.

4. Computer Vision

- Enables AI to analyze and interpret images and videos.
- Example: Medical image diagnosis, where AI detects diseases in X-rays and MRIs.

Applications of AI

AI is used across multiple industries to enhance efficiency and decision-making.

1. Healthcare

- AI assists in disease diagnosis, drug discovery, and personalized treatment.
- Example: IBM Watson Health, which analyzes medical data to help doctors diagnose diseases.

2. Finance

- AI predicts stock market trends and detects fraudulent transactions.
- Example: Robo-advisors like Betterment and Wealthfront.

3. Manufacturing

- AI-powered robots enhance production efficiency.
- Example: Tesla’s automated car assembly lines.

4. Education

- AI provides personalized learning experiences.
- Example: Duolingo, which adapts lessons based on users' learning speeds.

5. Automotive (Self-Driving Cars)

- AI powers autonomous vehicles.
- Example: Tesla’s Autopilot and Waymo’s self-driving taxis.

6. Entertainment

- AI creates and suggests content.
- Example: Spotify, YouTube recommendations.

Applied AI

- Refers to the practical implementation of AI to solve real-world problems in various industries.

- Unlike theoretical AI, which focuses on research and developing new AI models, applied AI leverages existing AI techniques such as ML, deep learning, and NLP to enhance business operations, improve efficiency, and automate tasks.

Key Features of Applied AI

1. Real-World Problem Solving – AI is used to address specific challenges in industries like healthcare, finance, manufacturing.
2. Practical Implementation – Uses AI technologies such as machine learning, NLP, and computer vision in everyday applications.
3. Data-Driven Decision Making – AI systems analyze large datasets to make predictions and optimize processes.
4. Automation & Efficiency – Reduces human effort and increases accuracy in various operations.

Examples of Applied AI in Different Industries

1. Healthcare - Medical Diagnosis, Virtual Health Assistants
2. Finance - Fraud Detection, Robo-Advisors
3. Manufacturing - Predictive Maintenance, Quality Control
4. Retail & E-Commerce - Personalized Shopping Recommendations, Chatbots for Customer Support
5. Automotive - Self-Driving Cars, Driver Assistance Systems
6. Education - Personalized Learning, AI-Based Exam Proctoring
7. Agriculture - Crop Monitoring with AI Drones, Automated Irrigation Systems
8. Smart Cities & Infrastructure - AI-Powered Traffic Control, AI for Public Safety

History of AI (1)

1. Early Concepts of AI (Before 1950s)

Ancient & Philosophical Foundations

- Greek Mythology: Concepts of artificial beings, such as Talos, a bronze automaton.
- Philosophy: Mathematicians and philosophers like Aristotle and Ramon Llull explored logic-based decision-making.

- Mechanical Automata: Early mechanical devices, like Leonardo da Vinci's robotic knight (1495), demonstrated the first attempts at automation.

2. Birth of AI (1950s - 1960s)

Key Developments:

- 1950 – Alan Turing: Proposed the Turing Test, a benchmark to determine if a machine can exhibit human-like intelligence.
- 1956 – Dartmouth Conference: John McCarthy, Marvin Minsky, and others coined the term *Artificial Intelligence*, marking AI as a distinct field.
- 1958 – Perceptron (Frank Rosenblatt): The first neural network model for machine learning.

AI Systems Developed:

- Early AI programs like Logic Theorist (1956) and General Problem Solver (1957) attempted human-like reasoning.
- ELIZA (1966): An early chatbot mimicking human conversation.

History of AI (2)

AI Boom and Challenges (1970s - 1980s)

- Expert Systems: AI was used in specialized domains like medical diagnosis (MYCIN) and business decision-making.
- 1973 – AI Winter (First Decline): Funding and interest in AI decreased due to high expectations and slow progress.

AI Challenges:

- Computational Limitations: Lack of hardware to process AI algorithms efficiently.
- High Costs: AI research required expensive resources, limiting real-world applications.

Machine Learning & Neural Networks (1990s - 2000s)

- Revival of AI: Increased computing power and improved algorithms revived interest in AI.
- 1997 – Deep Blue: IBM's AI defeated world chess champion Garry Kasparov, proving AI's potential.
- 1998 – Support Vector Machines (SVM): Enhanced machine learning models.

- 2000s – AI in Industry: AI was applied in speech recognition (Dragon) and recommendation systems (Amazon, Netflix).

History of AI (3)

Deep Learning & Big Data Era (2010s - Present)

- 2011 – IBM Watson: AI defeated human champions in *Jeopardy!*, showcasing NLP capabilities.
- 2012 – Deep Learning Breakthrough: Geoffrey Hinton's team developed AlexNet, a deep neural network that revolutionized computer vision.
- 2016 – AlphaGo: Google's DeepMind AI defeated world Go champion Lee Sedol, demonstrating AI's ability in complex strategic games.
- 2018 – GPT-2: AI-generated text with human-like fluency.
- 2024 – GPT-4: Advanced NLP model capable of conversation, text generation, and problem-solving.

Modern AI Applications:

- Self-driving cars (Tesla Autopilot)
- AI-powered medical diagnosis (IBM Watson, Google's DeepMind)
- Virtual assistants (Siri, Alexa, Google Assistant)
- AI-driven creativity (DALL·E, ChatGPT)

What is an Expert System?

- An Expert System (ES) is a computer-based application that mimics human expert decision-making in a specific domain.
- It uses a knowledge base and inference engine to solve complex problems, providing advice or decisions like a human expert.

Key Components of an Expert System

1. Knowledge Base – Stores facts, rules, and expert knowledge about a specific domain.
2. Inference Engine – Applies logical rules to analyze and infer conclusions from the knowledge base.
3. User Interface – Allows users to interact with the system (e.g., asking questions and receiving answers).

Difference Between Expert Systems and AI

Difference Between Expert Systems and AI

Feature	Expert System	Artificial Intelligence (AI)
Focus	Solves domain-specific problems using predefined rules	Learns and adapts to various tasks using data
Learning Ability	No learning (static knowledge base)	Learns from experience (ML, Deep Learning)
Decision Making	Rule-based (if-then logic)	Data-driven (pattern recognition, neural networks)
Adaptability	Fixed knowledge, needs manual updates	Can improve automatically with new data
Examples	MYCIN (Medical Diagnosis), DENDRAL (Chemical Analysis)	ChatGPT, DeepMind AlphaGo, Tesla Autopilot

Practical example of Expert Systems

Scenario: Medical Diagnosis System (MYCIN)

Imagine a medical expert system that helps doctors diagnose bacterial infections. Here's how it works:

1. User (Doctor) Inputs Symptoms
    a. Example: "Patient has fever, cough, sore throat, and difficulty breathing."
2. Inference Engine Matches Symptoms to Rules
    a. Rule 1: If fever + cough + sore throat = Possible flu
    b. Rule 2: If fever + cough + breathing issues = Possible pneumonia
3. Expert System Suggests a Diagnosis
    a. Output: "Based on the symptoms, the patient may have pneumonia. Recommended tests: X-ray, blood test."
4. Human Expert (Doctor) Makes Final Decision
    a. The doctor reviews the suggestion and confirms or modifies the diagnosis.

History of Expert Systems (1)

1. Early Foundations (1950s - 1960s)

Influence of Artificial Intelligence (AI)

- In the 1950s, AI research was focused on developing problem-solving and logical reasoning programs.
- Alan Turing (1950) proposed the Turing Test, which inspired early AI development.

Introduction of Rule-Based Systems

- AI researchers Newell and Simon created the General Problem Solver (1959), which laid the groundwork for expert systems.
- However, it was too generalized and couldn't mimic domain-specific expert reasoning.

History of Expert Systems (2)

2. The Birth of Expert Systems (1970s)

During the 1970s, AI research shifted towards specialized problem-solving systems, giving rise to expert systems.

Key Developments:

1. DENDRAL (1965) – First Expert System
    a. Developed at Stanford University by Edward Feigenbaum, Joshua Lederberg, and Bruce Buchanan.
    b. Designed to analyze chemical compounds and determine their molecular structure.
    c. First system to mimic the expertise of human chemists.
2. MYCIN (1972) – Medical Expert System
    a. Developed at Stanford University by Edward Shortliffe.
    b. Used to diagnose bacterial infections and recommend antibiotics.
    c. Worked based on if-then rules (e.g., "If the patient has fever and cough, consider pneumonia").
    d. Introduced certainty factors (confidence levels for decisions).
3. PROSPECTOR (1976) – Geology Expert System
    a. Used to identify potential mineral deposits.
    b. Helped geologists find a molybdenum deposit, which was later confirmed.

History of Expert Systems (3)

3. Commercialization and Peak of Expert Systems (1980s)

The 1980s saw a boom in expert systems, as companies realized their commercial potential.

Key Developments:

1. Rise of Business Expert Systems
    a. Companies like IBM, DEC, and Xerox adopted expert systems for decision-making.
    b. Used in areas like customer service, finance, and manufacturing.
2. Introduction of Rule-Based Programming Languages
    a. PROLOG (1972) and LISP became popular for building expert systems.
    b. Special AI shells like CLIPS (1985) made it easier to develop expert systems.
3. XCON (R1) – Digital Equipment Corporation (DEC)
    a. An expert system used to configure computer orders.
    b. Saved DEC millions of dollars annually by automating hardware configurations.
4. Banking and Finance Applications
    a. Expert systems were used in fraud detection, loan approvals, and stock market predictions.
    b. Example: FICO (Fair Isaac Corporation) used expert systems for credit scoring.

History of Expert Systems (4)

4. Decline and AI Winter (1990s)

Despite their success, expert systems began to decline in the 1990s due to several challenges.

Reasons for Decline:

1. High Development Costs
    a. Expert systems required manual knowledge encoding, making them expensive and time-consuming to build.
2. Scalability Issues
    a. Static rule-based systems couldn't learn or adapt to new situations.
    b. Required frequent updates, which made maintenance difficult.
3. Emergence of Machine Learning (ML)
    a. Machine learning and neural networks offered data-driven learning, unlike expert systems that relied on manually defined rules.

    b. ML models like decision trees and neural networks performed better in complex scenarios.

4. Computing Limitations

    a. Expert systems required significant computing power, which was not as advanced in the 1990s.

History of Expert Systems (5)

5. Evolution into Modern AI Systems (2000s - Present)

Although traditional expert systems declined, their concepts evolved into modern AI systems.

How Expert Systems Influenced AI Today:

1. Hybrid AI Systems

    a. Modern AI systems combine rule-based reasoning with machine learning.

    b. Example: IBM Watson uses both expert knowledge and deep learning for medical diagnosis.

2. Chatbots and Virtual Assistants

    a. Early expert systems influenced today's AI-powered chatbots (e.g., Siri, Alexa, ChatGPT).

    b. Example: ChatGPT can provide medical or legal advice, similar to expert systems like MYCIN.

3. AI in Healthcare & Finance

    a. AI-driven systems like DeepMind Health use expert knowledge + ML for advanced medical diagnosis.

    b. AI-powered financial systems use expert-system-inspired decision-making for risk assessment and fraud detection.

History of Expert Systems (6)

6. The Future of Expert Systems

Although traditional expert systems are rarely used, modern AI integrates expert knowledge with deep learning and NLP.

Future Trends:

1. Explainable AI (XAI)
    a. AI models will become more transparent, like expert systems, allowing better human understanding of AI decisions.
    b. Unlike traditional black-box AI models, XAI allows users to see and understand how an AI model reaches a conclusion.
2. AI-Powered Decision Support Systems
    a. AI-powered Decision Support Systems (DSS) assist humans in making better decisions by analyzing data and providing recommendations. 🔍 These systems do not replace human decision-makers but enhance their expertise.

3. Edge AI & IoT
    - Expert systems may be integrated into IoT devices for real-time decision-making (e.g., smart manufacturing, self-driving cars).

Traditional Interprétable Models vs. XAI

Aspect	Traditional Interpretable Models (Decision Trees, Logistic Regression, k-NN)	Explainable AI (XAI) Techniques (SHAP, LIME, Counterfactuals, PDP, Attention Maps)
Interpretability	**Inherently interpretable** (model structure is clear).	**Post-hoc explanations** (after decision is made).
Complexity	Works with **structured, tabular data**.	Works with **complex AI (deep learning, NLP, image processing, neural networks, black-box AI).**
Transparency	You can see **which features influence predictions** directly.	Uses **visualization and mathematical techniques** to explain **black-box AI models**.
Example Model	**Decision Trees, Logistic Regression, k-NN**	**Deep Learning (Neural Networks), Random Forest, XGBoost, LSTMs, Transformers**
Limitation	Works well for **small datasets** and **simple rules** but struggles with **big data & unstructured data** (images, text, video).	Required for **AI-driven automation, self-driving cars, NLP, facial recognition**, etc.

Expert Systems – Foundations & Applications

1. Foundations of Expert Systems

Expert Systems are a branch of AI designed to mimic human expertise in specific domains. They solve problems, provide recommendations, and assist in decision-making.

1.1 Definition of an Expert System

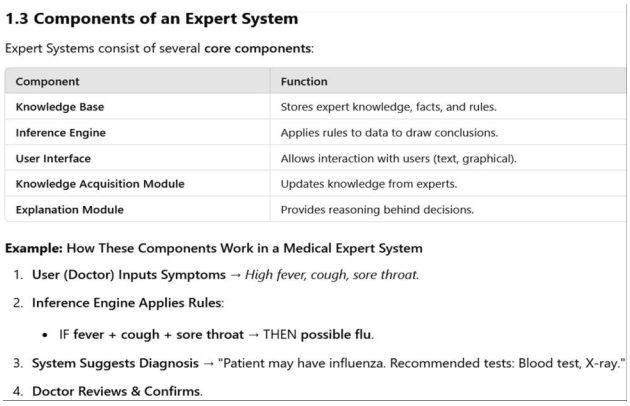
An Expert System (ES) is a computer program that simulates human expertise in a particular field using rule-based reasoning and knowledge representation.

- Example: A medical expert system can help doctors diagnose diseases by following predefined medical rules.

1.2 Key Characteristics of Expert Systems

1. Domain-Specific Knowledge – Focuses on a specific area (e.g., medical, finance, engineering).
2. Rule-Based Processing – Uses *IF-THEN* rules to make decisions.
3. Decision Support – Assists, but does not replace, human experts.
4. Reasoning Mechanism – Uses logical inference to analyze problems.
5. Explanation Capability – Can justify its decisions based on rules.

Components of an Expert System



1.3 Components of an Expert System

Expert Systems consist of several **core components**:

Component	Function
Knowledge Base	Stores expert knowledge, facts, and rules.
Inference Engine	Applies rules to data to draw conclusions.
User Interface	Allows interaction with users (text, graphical).
Knowledge Acquisition Module	Updates knowledge from experts.
Explanation Module	Provides reasoning behind decisions.

Example: How These Components Work in a Medical Expert System

1. **User (Doctor) Inputs Symptoms** → *High fever, cough, sore throat.*
2. **Inference Engine Applies Rules**:
   - IF **fever + cough + sore throat** → THEN **possible flu**.
3. **System Suggests Diagnosis** → "Patient may have influenza. Recommended tests: Blood test, X-ray."
4. **Doctor Reviews & Confirms**.

How Expert Systems Work

Knowledge Representation: Converts expert knowledge into a structured format (rules, logic, databases).

1. Inference Mechanism: Uses logical reasoning to process inputs.
2. Decision-Making Process: Applies rules and provides an output.

Example: Rule-Based System for Loan Approval

- IF credit score > 700 AND stable income THEN approve loan.
- IF credit score < 600 AND past defaults THEN reject loan.

Applications of Expert Systems

Expert Systems have widespread use across various industries. Below are major applications with examples.

2.1 Expert Systems in Healthcare

Expert Systems assist in medical diagnosis, treatment recommendations, and drug discovery.

Examples:

- MYCIN (1970s) – Diagnosed bacterial infections.
- IBM Watson Health – Analyzes medical data for disease detection.
- Clinical Decision Support Systems (CDSS) – Assists doctors in prescribing treatments.
- Case Study: IBM Watson for Cancer Diagnosis
- How It Works: Uses medical records and AI to suggest the best cancer treatments.
- Impact: Improves accuracy and reduces diagnosis time.

Expert Systems in Finance

Expert Systems help in fraud detection, risk assessment, and investment management.

Examples:

- FICO Score System – Determines creditworthiness.
- Algorithmic Trading Systems – Uses AI to predict stock market trends.
- Fraud Detection Systems – Identifies suspicious banking transactions.
- Case Study: Visa's AI-Based Fraud Detection
- How It Works: Uses AI and expert rules to detect fraudulent transactions in real time.
- Impact: Saves millions by preventing fraud.

Expert Systems in Manufacturing

Used for quality control, predictive maintenance, and production planning.

Examples:

- XCON (DEC) – Configured computer hardware orders.
- AI-Powered Robotics – Assists in automated manufacturing.
- Predictive Maintenance Systems – Prevents machinery failures by predicting breakdowns.
- Case Study: General Electric's AI-Based Maintenance
- How It Works: Uses sensor data to predict machine failures.
- Impact: Reduces downtime and saves costs.

Expert Systems in Agriculture

Expert Systems optimize crop management, pest control, and weather prediction.

Examples:

- AgriSmart – AI-based crop disease detection.
- FarmBot – Uses AI to optimize irrigation and fertilization.
- Drone-Based Crop Monitoring – AI-powered drones analyze soil and crop health.
- Case Study: Smart Farming with AI
- How It Works: AI sensors analyze soil nutrients and weather.
- Impact: Increases productivity while reducing waste.

Expert Systems in Cybersecurity

Helps in detecting cyber threats and automating security measures.

Examples:

- Intrusion Detection Systems (IDS) – Monitors networks for threats.
- Antivirus and Malware Detection – Uses expert rules to detect suspicious activity.
- AI-Powered Threat Analysis – Predicts and prevents cyber attacks.
- Case Study: AI-Powered Network Security
- How It Works: Uses AI to monitor and detect cyber threats.
- Impact: Enhances security in large organizations.

Expert Systems in Education

Used for personalized learning, tutoring, and automated grading.

Examples:

- AI Tutors – Adaptive learning systems (e.g., Duolingo).
- Automated Grading Systems – AI-based exam assessment.
- Career Counseling Expert Systems – Suggests suitable courses based on student performance.
- Case Study: AI in Online Learning

- How It Works: AI adapts difficulty levels based on student responses.
- Impact: Enhances personalized learning experiences.

Expert Systems in Business & Customer Support

Used for decision-making, automation, and customer service.

Examples:

- Chatbots (AI Virtual Assistants) – Customer support automation.
- Supply Chain Optimization – AI for demand forecasting.
- AI-Powered CRM (Customer Relationship Management) – Enhances customer interactions.
- Case Study: AI Chatbots in Banking
- How It Works: AI chatbots handle customer queries.
- Impact: Reduces wait times and improves service.

Phases in Building an Expert System

1. Problem Identification
2. Knowledge Acquisition (Extracting Expert Knowledge)
3. Knowledge Representation (Storing Knowledge in a Structured Form)
4. Designing the Inference Engine (Decision-Making Logic)
5. User Interface (I/O) Development
6. Testing & Validation
7. Deployment & Maintenance

Problem Identification

- Identify the problem domain (What kind of expertise is needed?).
- Define the scope of the expert system.
- Determine if an expert system is the best solution.

Example:

If a hospital wants to develop an expert system to help diagnose diseases, they need to:

✓ Identify the types of diseases it will diagnose.

✓ Define which symptoms, tests, and risk factors will be considered.

✓ Determine if an expert system can improve accuracy in diagnosis.

Knowledge Acquisition (Extracting Expert Knowledge)

- Collect expert knowledge from domain specialists (doctors, engineers, financial analysts).
- Use interviews, case studies, books, or past case records.
- Structure the knowledge into facts, rules, and relationships.

Example:

For a medical expert system, a doctor (expert) may provide the following knowledge:

Rule 1: If a patient has fever, cough, sore throat → Possible flu.

Rule 2: If fever + cough + breathing issues → Possible pneumonia.

This knowledge will be later converted into a rule-based system.

3. Knowledge Representation

- Organize knowledge in a format the computer system can understand.
- Use rules, frames, semantic networks, or ontologies.

Methods of Knowledge Representation:

1. Rule-Based Representation (IF-THEN Rules)
    a. Example:
        i. IF fever AND sore throat → THEN Flu is likely.
        ii. IF chest pain AND shortness of breath → THEN Heart disease is possible.
2. Frame-Based Representation

Example:

        i. A "Disease" frame might have attributes like name, symptoms, treatment.

2. Semantic Networks (Graph-Based Representation)

Example:

        i. "Flu" is linked to symptoms like "cough" and "fever."

Designing the Inference Engine

- Develop the reasoning mechanism that allows the expert system to draw conclusions from the knowledge base.
- The Inference Engine applies rules and logic to given facts.

Types of Reasoning in the Inference Engine

1. Forward Chaining (Data-Driven Reasoning)
    a. Starts with known facts and applies rules to reach a conclusion.
    b. Example:
        i. Symptoms: Fever, sore throat → Rule: Flu is likely.
    c. Used in diagnostic systems (medical diagnosis, fraud detection, etc.).
2. Backward Chaining (Goal-Driven Reasoning)
    a. Starts with a goal (hypothesis) and works backward to verify if conditions match.
    b. Example:
        i. AI asks, "Does the patient have a fever?" to confirm flu diagnosis.
    c. Used in troubleshooting systems (engineering fault detection, legal expert systems.).

User Interface Development

- Design an interface where users interact with the expert system.
- Users input data (symptoms, financial details, etc.), and the system provides advice.

🔷 Example:

A Doctor using a Medical Expert System will: ✓ Enter patient symptoms into the system.

✓ The system analyzes data using the Inference Engine.

✓ It displays the possible disease and suggests further tests/treatment.

Types of Interfaces:

Text-Based – Simple question-answer format.

Graphical (GUI) – Visual representations with menus, buttons, images.

Testing & Validation

- Test if the expert system provides accurate and reliable outputs.
- Validate against expert opinions to ensure correctness.

🔷 Example:

- A Medical Expert System is tested using past patient records to check if the diagnosis is correct.
- A Financial Expert System is tested by predicting stock market trends.
- If errors are found, the knowledge base is refined to improve accuracy.

Deployment & Maintenance

- Deploy the expert system for real-world use.
- Regularly update the knowledge base with new data.

Example:

✓ A Legal Expert System needs updates when new laws are introduced.

✓ A Medical Expert System is updated with new diseases and treatments.

Phases in Building an Expert System

Phase	Description	Example
1. Problem Identification	Define the problem the system will solve.	Identify if an AI system can assist in **medical diagnosis**.
2. Knowledge Acquisition	Gather information from human experts.	Collect expert **medical rules from doctors**.
3. Knowledge Representation	Convert expert knowledge into a structured form.	Use **IF-THEN rules, frames, or semantic networks**.
4. Inference Engine Development	Implement reasoning mechanisms to analyze data.	Use **Forward Chaining** to diagnose diseases.
5. User Interface Design	Develop an interactive system for users.	A **GUI where doctors input patient symptoms**.
6. Testing & Validation	Check accuracy against expert knowledge.	Verify predictions against real medical cases.
7. Deployment & Maintenance	Deploy the system and update knowledge.	Update **medical guidelines & treatments**.

Architecture of Expert System

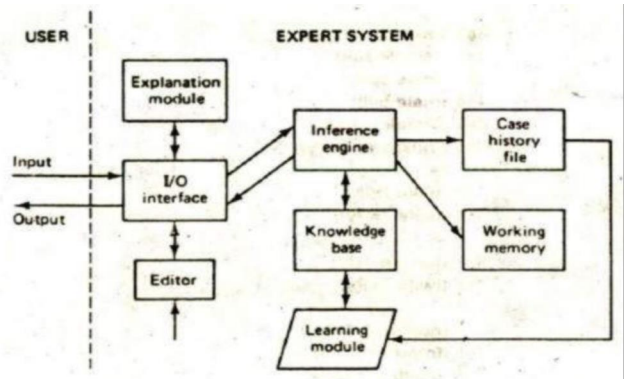


Figure 15.1 Components of a typical expert system.

Architecture of Expert Systems (1)

1. I/O Interface: Interface through which the user interacts with the expert system. It allows users to input queries and receive outputs in a user-friendly manner.

It acts as a bridge between the user and the system, taking user inputs and displaying the results from the system.

2. Editor: Used for entering, modifying, or updating the rules and facts in the expert system.

It can be used by knowledge engineers to update the system's knowledge base.

3. Explanation Module: Provides the system's reasoning to the user by explaining how a particular solution or decision was reached.

It is important for providing transparency, as users can understand why the system gave a specific recommendation.

4. Inference Engine: Core of the expert system.

The inference engine applies the rules from the knowledge base to the facts provided by the user to derive conclusions or make decisions.

It uses forward or backward chaining to match conditions in the knowledge base with the facts.

Architecture of Expert Systems (2)

5. Knowledge Base: Contains facts and rules as memory.

Facts are specific pieces of information (e.g., "Bob is male").

Rules are condition-action statements (e.g., "If X is male and X has a child Y, then X is the father of Y").

The knowledge base stores domain-specific knowledge necessary for problem-solving.

6. Learning Module: Helps the expert system update and adapt over time by learning new facts or rules from past cases.

It can modify or add new information to the knowledge base, making the system smarter over time.

7. Working Memory: Temporary storage where the system keeps facts or intermediate results during processing.

It helps in storing information required to solve a particular problem during inference.

8. Case History File: Stores previous cases or instances the system has worked on.

Helps the system reuse information from previous cases, improving its problem-solving ability over time.

9. Learning Module: Helps improve the expert system by allowing it to "learn" from new data, cases, or inputs.

This module allows the system to evolve and improve its knowledge over time.

Rule-Based Expert Systems

A Rule-Based Expert System (RBES) is a type of Artificial Intelligence (AI) system that mimics human decision-making using a set of rules (IF-THEN statements).

How It Works?

✓ Uses IF-THEN rules to make decisions.

✓ Has a knowledge base (facts + rules).

✓ Uses an inference engine to apply rules and reach conclusions.

Example:

- Rule: "IF it is raining, THEN carry an umbrella."
- Input: It is raining.
- Conclusion: Carry an umbrella.

This simple logic can be scaled up to create expert systems for medical diagnosis, troubleshooting, finance, etc.

Components of a Rule-Based Expert System

Component	Function
Knowledge Base	Stores **facts & IF-THEN rules**.
Inference Engine	**Applies rules** to facts to reach conclusions.
User Interface	Allows users to **input data & receive explanations**.

Example: Medical Diagnosis System

Knowledge Base:

- Fact 1: Fever + Cough → Possible Flu.
- Fact 2: Chest Pain + Breathing Issues → Possible Pneumonia.

Inference Engine:

- If the patient has a fever & cough, THEN diagnose flu.

User Interface:

- The doctor inputs symptoms, and the AI suggests a diagnosis.

How Rule-Based Expert Systems Work?

A Rule-Based Expert System follows four steps:

Step 1: Knowledge Acquisition

✓ Experts provide knowledge.

✓ Converts expertise into IF-THEN rules.

Step 2: Knowledge Representation

✓ Stores facts & rules in a structured format.

✓ Uses rule-based logic (IF-THEN statements).

Step 3: Inference Engine Processing

✓ Matches facts to rules.

✓ Uses Forward Chaining or Backward Chaining to derive conclusions.

Step 4: User Interaction

✓ The user provides input (e.g., patient symptoms).

✓ The system applies rules and suggests a conclusion/diagnosis.

Teaching Rule-Based Expert Systems

🔹 **Step 1: Define the Rules**

We create **IF-THEN rules** based on **medical knowledge**:

Rule	Condition	Conclusion
Rule 1	If **fever** and **cough**	Then **diagnose flu**
Rule 2	If **chest pain** and **breathing issues**	Then **diagnose pneumonia**
Rule 3	If **flu symptoms** + **runny nose**	Then **diagnose common cold**

Step 2: Apply Rules to Input Data

- If a patient enters symptoms, the system applies rules:

Patient Input:

- Symptoms: Fever, Cough AI Diagnosis: Flu (Based on Rule 1).

Patient Input:

- Symptoms: Chest Pain, Breathing Issues
- AI Diagnosis: Pneumonia (Based on Rule 2).

Forward Chaining vs. Backward Chaining

Forward Chaining (Data-Driven)

- Starts with facts and applies rules to reach a conclusion.

Example: "Fever + Cough → Flu."

Backward Chaining (Goal-Driven)

- Starts with a goal (flu) and works backward to verify facts.

Example: "Does the patient have a fever?" → If yes, check cough.

Applications of Rule-Based Expert Systems

✓ Medical Diagnosis – AI-powered health assistants.

✓ Troubleshooting Systems – AI helps fix technical issues.

✓ Fraud Detection – AI detects fraudulent transactions.

✓ Legal AI – AI assists lawyers in case analysis.

Expert Systems vs. Traditional Systems (1)

Feature	Expert System	Traditional System
Definition	AI-based system that mimics **human expert decision-making**.	Pre-programmed system that follows **fixed instructions**.
Decision-Making	Uses **IF-THEN rules, reasoning, and inference engines**.	Uses **fixed logic and predefined algorithms**.
Adaptability	Can be **updated with new knowledge**.	Hardcoded logic, **requires reprogramming** for changes.
Learning Ability	**Limited learning** (some systems use AI for updating).	**No learning capability**, follows strict commands.
Reasoning Approach	Uses **Forward & Backward Chaining** to reach conclusions.	Executes **predefined workflows** without reasoning.
Explanation Ability	Can provide **reasoning** for its conclusions (**XAI concepts**).	Cannot **explain why it made a decision**.
Handling Uncertainty	Uses **Probability, Fuzzy Logic, and Heuristics** to deal with **uncertain data**.	Works only with **structured and precise inputs**.

Expert Systems vs. Traditional Systems (2)

User Interaction	Interacts using **natural language processing (NLP), expert dialogues, or chatbots**.	**Menu-driven or command-based interactions**.
Examples	**Medical Diagnosis (IBM Watson Health), AI Legal Advisor, Fraud Detection Systems**.	**Bank transaction system, ATM software, e-commerce platforms**.
Flexibility	Can be **applied across different industries** by modifying its knowledge base.	Designed for **specific tasks** only.
Development Time	Requires **expert knowledge gathering, rule encoding, and validation**.	Faster to develop as it follows **predefined logic**.
Complexity Handling	Can handle **complex, unstructured problems**.	Works well with **structured, routine problems**.
Industry Applications	Healthcare, Law, Cybersecurity, Finance, Manufacturing.	Basic business automation, banking, traditional computing.

Blackboard System

A Blackboard System is an AI-based problem-solving architecture where multiple independent experts (knowledge sources) contribute to solving a problem by writing and updating information on a shared "blackboard."

How Does It Work?

- The blackboard is like a shared workspace where different AI modules (agents) write and read information.
- The system gradually refines solutions as each expert contributes.
- Used in complex problem-solving, where multiple approaches must be combined.

Real-Life Analogy:

Students to explain about a topic like fuzzy logic on blackboard.

Component	Function
Blackboard (Workspace)	A **shared memory space** where knowledge is written, updated, and read.
Knowledge Sources (KS)	Independent **AI experts** that contribute **specialized knowledge**.
Control Mechanism (Scheduler)	Manages **which expert contributes next** based on current progress.

How Blackboard Systems Work

A Blackboard System follows four phases:

🔷 Step 1: Problem Definition

- Define the goal and what knowledge sources (KS) are needed.
- Example: Understanding speech in an AI Voice Assistant.

🔷 Step 2: Blackboard Initialization

- The system starts with an empty blackboard.
- Each knowledge source waits until relevant data appears.

🔷 Step 3: Knowledge Sources Contribute

- Experts (KS) update the blackboard as more data becomes available.
- The control mechanism decides which expert works next.

🔷 Step 4: Solution Refinement

- The system iteratively refines solutions until an answer is found.

Real-World Applications of Blackboard Systems (1)

🛠️ Example 1: AI-Based Speech Recognition (Siri, Alexa, Google Assistant)

💡 **Problem:** Understanding human speech using AI.
💡 **Solution:** A Blackboard System with **multiple knowledge sources**.

Knowledge Source	Task Performed
Acoustic Processing	Identifies **speech sounds** (phonemes).
Language Model	Determines **probable words** based on sounds.
Grammar Rules	Ensures **words form a correct sentence**.
Context Analysis	Understands **meaning based on conversation**.

🔹 **Process:**
1️⃣ The **acoustic processor** identifies "H-E-L-L-O".
2️⃣ The **language model** suggests "Hello" or "Help".
3️⃣ The **grammar model** confirms "Hello" is correct.
4️⃣ The **context model** determines if "Hello" is a greeting.

✅ **Final Output:** AI assistant **understands speech and responds correctly**.

Real-World Applications of Blackboard Systems (2)

🛠️ Example 3: Self-Driving Cars

💡 **Problem:** How does an AI **navigate traffic safely**?
💡 **Solution:** The **Blackboard System** combines **different AI experts** to make driving decisions.

Knowledge Source (KS)	Task Performed
Sensor Processing Module	Analyzes camera, LiDAR, and radar data.
Traffic Rule Module	Ensures car follows road laws.
Obstacle Detection	Identifies pedestrians, cars, and roadblocks.
Path Planning Module	Plans the safest route based on real-time data.

🔹 **Process:**
1️⃣ The **sensor module** detects a **pedestrian crossing**.
2️⃣ The **traffic rule module** checks if the car **must stop**.
3️⃣ The **path planner** recalculates a **safe driving path**.

✅ **Final Output:** The **self-driving car stops safely** to avoid the pedestrian.

Real-World Applications of Blackboard Systems (3)

Feature	Blackboard System	Rule-Based Expert System
Approach	Collaborative problem-solving **(multiple AI experts working together)**.	Uses **predefined IF-THEN rules**.
Handling Complexity	**Handles complex, multi-step problems** with different knowledge sources.	Works best for **specific, structured decisions**.
Example Use Case	**Self-driving cars, speech recognition, medical AI.**	**Medical diagnosis, legal AI, fraud detection.**
Decision Process	Iteratively refines solutions **over time**.	Uses **fixed rules** to reach an answer quickly.

Advantages of Blackboard Systems

Handles Complex Problems – Ideal for real-world AI applications.

Flexible & Modular – New knowledge sources can be added easily.

Multiple Experts Collaborate – Works like a team solving a problem together.

Truth Maintenance System (TMS)

A Truth Maintenance System (TMS) is an AI reasoning mechanism that manages, updates, and corrects facts in an Expert System as new information becomes available.

💡 In simple terms:

- A TMS keeps track of what is true and false in an expert system.
- When new information arrives, it updates old facts accordingly.
- It avoids contradictions and ensures logical consistency.

🔷 Why is TMS Needed?

- Facts change over time (e.g., weather forecasts, medical diagnoses, financial trends).
- AI systems must update beliefs dynamically to make better decisions.

📌 Example:

- An AI system believes: "John has a fever, so he may have the flu."
- A new test result shows that John has malaria, not the flu.
- TMS updates the system → "John does NOT have flu but has malaria."

👉 TMS helps AI avoid outdated or incorrect conclusions!

How Does a Truth Maintenance System Work?

A **TMS operates in three key steps**:

Step	Process
1 **Assumption & Fact Storage**	Stores facts based on available knowledge.
2 **Justification Tracking**	Keeps track of WHY a fact is believed to be true.
3 **Updating Knowledge**	If new information contradicts old facts, TMS updates beliefs.

📌 **Key Components of a TMS**

Component	Function
Knowledge Base	Stores facts & assumptions.
Inference Engine	Uses rules to derive conclusions.
Justification Structure	Tracks **why a fact is believed to be true**.
Contradiction Detector	Detects **inconsistent facts** & updates them.

Types of Truth Maintenance Systems

There are **two major types of TMS**:

TMS Type	How It Works	Example
Justification-Based TMS (JTMS)	Maintains a **list of justifications** for facts and removes facts when justifications change.	AI believes "**John is sick because he has a fever.**" If John **tests negative for flu**, the system **removes the flu diagnosis**.
Assumption-Based TMS (ATMS)	Allows AI to track **multiple possible scenarios** and switch between them as needed.	AI in **self-driving cars** considers multiple **routes** and updates based on **traffic conditions**.

Example 1: Fraud Detection in Banking (TMS)

💡 **Problem:** Banks use AI to **detect fraud** but need to update conclusions if transactions are later verified as **legitimate**.
💡 **Solution: TMS updates fraud detection systems when new evidence appears.**

📌 **Scenario:**

- AI flags **a $10,000 transaction** as **fraudulent**.
- The customer later **confirms that they made the transaction**.
- **TMS updates the fraud status**:
  - ❌ Removes "Transaction is fraudulent".
  - ✅ Adds "Transaction is valid".
- The AI **unblocks the customer's account**.

✅ **Final Output: AI avoids false fraud alerts and improves accuracy.**

Example 2: AI in Robotics (TMS)

💡 **Problem:** A robot must **update its environment map when obstacles appear**.
💡 **Solution: TMS ensures the robot's AI dynamically adapts to changes.**

📌 **Scenario:**

- A **warehouse robot** plans to move **straight ahead**.
- **A box appears in its path**.
- **TMS updates the robot's knowledge**:
  - ❌ "Path is clear" is **removed**.
  - ✅ "Obstacle detected, take an alternate path" is **added**.
- The robot **chooses a different route**.

✅ **Final Output:** The **robot avoids the obstacle and reaches its destination safely.**

Shells and Tools in Expert Systems

Expert System Shells and Tools are pre-built frameworks or software platforms that allow users to develop Expert Systems (ES) without coding everything from scratch.

👉 Shells: Pre-programmed software that provides built-in inference engines and knowledge representation techniques.

👉 Tools: Programming languages or development environments used to build custom expert systems.

Feature	Expert System Shells	Expert System Tools
Definition	**Pre-built frameworks** with built-in knowledge bases, inference engines, and user interfaces.	**Programming languages, libraries, or AI development environments** for building expert systems from scratch.
Ease of Use	Easier to use, designed for **non-programmers**.	Requires **coding skills**, used by AI developers.
Customization	Limited to the features provided by the shell.	Fully customizable, users can build **any expert system**.
Examples	CLIPS, Jess, Exsys, Drools.	Python, LISP, Prolog, Java, Expert System APIs.
Best For	Quick development of expert systems **without deep programming knowledge**.	AI researchers & developers who want **full control over expert system design**.

What is a Framework?

Framework = Conductor & Sheet Music

- In an **orchestra**, the **conductor (framework)** organizes **musicians (developers)**.
- The **sheet music (rules)** ensures **everyone follows a structured approach**.
- Each musician **plays an instrument**, but they **follow the same system** to create a symphony.

🔹 **Real-World Example:**

- **Django = Classical music performance (structured, all parts pre-arranged).**
- **Flask = Jazz improvisation (loose structure, more flexibility).**

Framework = Car Chassis & Engine
Framework = Recipe Book

Why use Framework

🚀 Saves Time – No need to write everything from scratch.

⚙ Provides Structure – Helps organize code better.

🔧 Handles Complex Tasks – Frameworks manage networking, security, database handling, and more.

💡 More Reliable – Pre-tested and widely used by developers.

Category	Frameworks	Used For
Web Development	Django, Flask, FastAPI	Building websites, web apps, REST APIs
Machine Learning & AI	TensorFlow Extended (TFX), Keras, FastAI	AI, deep learning, data science
Game Development	Pygame, Panda3D	Creating 2D/3D games
GUI Development	Tkinter, PyQt, Kivy	Making desktop/mobile applications
Data Analysis	Kedro, Dask, Vaex	Processing and visualizing large-scale data
Cybersecurity & Hacking	Metasploit Framework, OpenVAS	Security analysis, penetration testing

Expert System Shells: How They Work

An Expert System Shell already includes:

1️⃣ A built-in Knowledge Base (stores rules & facts).

2️⃣ An Inference Engine (processes rules & makes decisions).

3️⃣ A User Interface (allows users to input data & get explanations).

🔷 Steps to Build an Expert System Using a Shell:

✓ Step 1: Define the knowledge base (rules & facts).

✓ Step 2: Use the built-in inference engine to process logic.

✓ Step 3: Create a user-friendly interface for data input & output.

Expert System Shell	Description	Used In
CLIPS (C Language Integrated Production System)	A widely used expert system shell for rule-based AI development.	Used in NASA, aerospace research, industrial automation.
Jess (Java Expert System Shell)	Java-based shell for integrating expert systems in enterprise applications.	Used in business decision-making, stock market prediction.
Exsys Corvid	Graphical expert system shell, no coding required.	Used in healthcare, legal AI, banking.
Drools (Business Rules Management System)	Rule-based AI system used in enterprise business applications.	Used in finance, insurance, supply chain management.

Expert System Tools: How They Work

Tools provide programming flexibility, allowing developers to build expert systems from scratch.

Expert System Tool	Description	Used In
Prolog (Logic Programming Language)	A logic-based AI language used to create expert systems.	Used in legal AI, medical AI, and robotics.
LISP (List Processing Language)	One of the oldest AI programming languages, used for building expert systems.	Used in early AI research, machine learning.
Python (AI & ML Development)	Provides libraries like Pyknow for expert systems.	Used in modern AI applications, chatbots, and medical diagnosis AI.

Python (Pyknow) to Build an Expert System

💡 **Scenario**: AI expert system for **loan approval.**

🔹 **Using Pyknow (Python Expert System Tool)**:

```python
from experta import *

class LoanApproval(KnowledgeEngine):
    @Rule(Fact(credit_score="high"), Fact(income="stable"))
    def approve_loan(self):
        print("Loan Approved!")

    @Rule(Fact(credit_score="low"))
    def reject_loan(self):
        print("Loan Rejected!")

engine = LoanApproval()
engine.reset()
engine.declare(Fact(credit_score="high"), Fact(income="stable"))
engine.run()
```

✅ **Output**: The expert system **approves a loan** if the credit score is high.

Shells vs. Tools

Scenario	Best Choice	Example
You want to quickly create an expert system without coding.	Expert System Shell	Use CLIPS, Jess, Exsys Corvid.
You need full customization & complex AI logic.	Expert System Tool	Use Python, Prolog, LISP.
You are building a business AI system.	Drools (Shell)	Use Drools for business rules management.
You are developing a legal AI system.	Prolog (Tool)	Use Prolog for logical reasoning.