UNIT IV: Artificial Intelligence and Expert System

1. Concept of Learning

1.1 Definition

Learning in AI is the **process by which an agent improves its performance** at tasks over time, by acquiring knowledge or skills from data, experience, or interaction, without being explicitly re-programmed for each scenario.

1.2 Types of Learning

1.2.1 Supervised Learning

- **Definition:** Learns a mapping from inputs to outputs using labeled examples.
- **Process:**
  1. **Training Phase:** Feed (input, correct output) pairs to a learning algorithm.
  2. **Model Construction:** Adjust internal parameters to minimize error.
  3. **Testing Phase:** Validate on unseen inputs to measure generalization.
- **Examples:** Linear regression, decision trees, support vector machines.

1.2.2 Unsupervised Learning

- **Definition:** Discovers patterns or groupings in unlabeled data.
- **Process:**
  1. **Cluster Analysis:** Group similar data points (e.g., K-means).
  2. **Dimensionality Reduction:** Find latent structure (e.g., PCA).
- **Examples:** Market segmentation, anomaly detection.

1.2.3 Reinforcement Learning

- **Definition:** Learns by **trial-and-error**, receiving feedback in the form of rewards or penalties.
- **Process:**
    1. **Agent–Environment Loop:** Agent takes action → environment returns reward + new state.
    2. **Policy Update:** Agent updates strategy to maximize cumulative reward.
- **Examples:** Game playing (AlphaGo), robotic control.

1.3 Learning Models & Algorithms

Model Type	Key Characteristics	Example Algorithms
Instance-based	Memorizes training instances	k-Nearest Neighbors
Model-based	Builds a global model	Linear/Logistic Regression
Neural Networks	Multi-layered nodes, gradient descent	MLP, CNN, RNN
Probabilistic	Encodes uncertainty	Naive Bayes, Bayesian Nets

1.4 Advantages & Limitations

- **Advantages:**
    - Adaptability to new data
    - Can uncover complex patterns
- **Limitations:**
    - Requires large, high-quality datasets
    - Risk of overfitting or underfitting

---

2. Knowledge Acquisition

2.1 Definition

The **process of gathering** domain expertise and transforming it into a formal knowledge base for AI systems.

2.2 Methods of Acquisition

1. **Manual Elicitation**
    - **Interviews & Workshops** with domain experts
    - **Protocol Analysis:** Experts think aloud while solving problems
2. **Automated Extraction**
    - **Text Mining:** Parsing manuals, reports
    - **Machine Learning:** Inducing rules from historical data
3. **Mixed-Mode**
    - **Interactive Tools** (e.g., knowledge editors) that blend expert input with automated suggestions

---

2.3 Knowledge Engineering Cycle

1. **Feasibility Study:** Assess problem complexity, ROI
2. **Knowledge Elicitation:** Capture raw expertise
3. **Knowledge Modeling:** Choose representation (rules, frames, ontologies)
4. **Implementation:** Populate knowledge base
5. **Verification & Validation:** Test against real cases
6. **Maintenance:** Update as domain evolves

---

3. Rote Learning

3.1 Definition

Storing exact facts or examples in memory without inference or generalization.

3.2 Process & Characteristics

- **Memorization:** Facts are added verbatim to the knowledge base.
- **Recall:** System matches input to stored facts for retrieval.

Pros	Cons
Simple to implement	No ability to handle novel cases
Fast lookup for exact matches	Knowledge base grows quickly
Predictable behavior	Zero inferential power

4. Discovery

4.1 Definition

Automatic or semi-automatic **generation of new knowledge** (rules, patterns) from data or by exploring model behavior.

4.2 Approaches

- **Rule Induction:** Generate IF–THEN rules from data (e.g., Quinlan's C4.5).
- **Genetic Algorithms:** Evolve rule sets by selection/crossover/mutation.
- **Clustering & Outlier Analysis:** Identify novel groupings or anomalies.

4.3 Example: Rule Induction Workflow

1. **Select Candidate Attributes**
2. **Partition Data** (e.g., decision tree splits)
3. **Formulate Rules** from leaves
4. **Prune** to remove over-specific rules

5. Analogy

5.1 Definition

Problem-solving by **mapping** structures or relationships from a known domain (source) to a new domain (target).

5.2 Structure-Mapping Theory (Gentner)

- **Base Domain:** Known situation with relational structure.
- **Target Domain:** New situation to be understood or solved.
- **Mapping:** Identify correspondences between base and target entities.

5.3 Example

- **Source:** Solar system (Sun-planets orbits)
- **Target:** Atom (Nucleus-electrons orbits)

By analogy, infer electron orbits behave like planetary orbits.

6. Concept of Expert System

6.1 Definition

A computer program that **emulates the decision-making** ability of a human expert using a structured knowledge base and inference engine.

7. Need for an Expert System

- **Expert Scarcity:** Capture and reuse rare expertise.
- **Consistency:** Provide uniform advice 24×7.
- **Cost Efficiency:** Reduce reliance on costly specialists.

- **Training & Documentation:** Serve as a learning tool.

---

8. Components of an Expert System

1. **Knowledge Base (KB):**
    - Facts, rules, frames, ontologies.
2. **Inference Engine (IE):**
    - **Forward Chaining:** Data-driven reasoning.
    - **Backward Chaining:** Goal-driven reasoning.
3. **Working Memory (WM):**
    - Temporary storage of facts during reasoning.
4. **User Interface (UI):**
    - Query input, explanation display.
5. **Explanation Facility:**
    - Traces reasoning steps (“Why?” and “How?”).
6. **Knowledge Acquisition Module:**
    - Tools for updating KB (semi-automated).

---

9. Categories of Expert Systems

Category	Description	Example
Rule-Based ES	Uses IF–THEN rules	MYCIN (medical diagnosis)
Frame-Based ES	Uses frames/objects with slots and inheritance	KEE (Knowledge Engineering Environment)
Fuzzy ES	Handles uncertainty via fuzzy sets	Fuzzy controllers in appliances
Case-Based ES	Solves new problems by retrieving similar past cases	CBR systems for legal advice
Model-Based ES	Uses models of system behavior (physical/functional)	Hardware fault diagnostic tools

10. Stages in Development of an Expert System

1. **Problem Definition & Scope**
2. **Feasibility Analysis** (technical, economic)
3. **Knowledge Elicitation** (expert interviews, document analysis)
4. **Knowledge Representation Design** (choose rule/frame/case)
5. **Prototype Implementation** (core rules + IE)
6. **Validation & Verification** (test on benchmark cases; expert review)
7. **User Interface Design** (ease of interaction, explanations)
8. **Deployment & Testing** (in real environment, pilot runs)
9. **Maintenance & Evolution** (update KB, refine rules)
10. **User Training & Documentation**