UNIT IV

First Order Logic:

First-Order Logic (FOL), also known as Predicate Logic, allows for more detailed and expressive reasoning.

Key Components of First-Order Logic:

1. **Objects:** These are the entities in the universe being discussed. Example: John, a car, a book.

2. **Predicates:** These describe properties of objects or relations between them. Predicates are functions that return true or false. Example: Person(John) (John is a person), Loves(Mary, John) (Mary loves John).

3. **Quantifiers:** These specify how many objects a statement refers to. There are two main quantifiers:

->**Universal Quantifier (∀):** Means "for all." It asserts that a statement is true for every object in the domain.
Example: $\forall x$ Person(x) → Mortal(x) (All people are mortal).
->**Existential Quantifier (∃):** Means "there exists." It asserts that there is at least one object for which the statement is true.
Example: $\exists y$ (Dog(y) ∧ Barks(y)) There exists a dog that barks

4. **Variables:** These represent objects in the domain of discourse. Example: x, y, z.

5. **Functions:** These map objects to other objects

Example: Father(John) (John's father), Age(John) (John's age).

6. **Logical Connectives:** These are used to combine predicates and form complex statements, just like in propositional logic:

∧ (and), ∨ (or), ¬ (not), → (implies), ↔ (if and only if).

Example of First-Order Logic Statement:

"Everyone who loves ice cream is happy," using FOL:

$$\forall x\ (Loves(x, IceCream) \rightarrow Happy(x))$$

Differences Between First-Order Logic and Propositional Logic:

Feature	Propositional Logic	FOL
What it deals with	Simple statements (true or false)	Objects and their relationships
Example	"It is raining."	Chases(Dog, Cat) – The dog is chasing the cat.
Variables	None	Uses variables like x, y for objects
Quantifiers	None	Uses ∀ (for all) and ∃ (there exists)
Relationships	Cannot show relationships	Can express relationships (e.g., "John is taller than Mary.")
Complexity	Simple and straightforward	More powerful, can handle detailed reasoning
Used for	Basic facts and logic circuits	Complex problems in math, AI, and reasoning

Inference In First Order Logic:

In First-Order Logic, inference is used to derive new facts **using sybstitution and Equality**

Substitution

Substitution in FOL is about replacing variables with constants or terms. When you see `F[a/x]`, it means replacing the variable `x` with the constant `a`.

Example: If `P(x)` means "x is a person," and you substitute `John` for `x`, you get `P(John)`, meaning "John is a person."

Equality

Equality helps show that two terms represent the same object.

Example: `Brother(John) = Smith` means John’s brother is the same person as Smith. Negation is used to show inequality: `x ≠ y` means `x` and `y` are different objects.

Inference Rules for Quantifiers

1. **Universal Generalization**: If something is true for any particular object, it is true for everything in the universe of discourse.

   Example: If "A byte contains 8 bits" is true for one byte, it's true for all bytes.

2. **Universal Instantiation**: From a general statement about all objects, you can infer a specific instance.

   Example: From "All people like ice cream," you can infer "John likes ice cream."

3. **Existential Instantiation**: If something exists, you can assume it exists with a specific constant that hasn’t been used yet.

   Example: If "There is a crown on someone's head," you can infer "Crown(K) is on John's head," where K is a new term.

4. **Existential Introduction**: If something is true for a specific individual, you can infer it’s true for someone.

   Example: If "Priyanka got good marks in English," you can say "Someone got good marks in English."

Generalized Modus Ponens

This is the core rule of inference in FOL. It says that if a general statement is true, and a specific instance of that statement is true, then you can conclude something new.

Example:

- Statement: "If someone is a king and greedy, they are evil."
- Fact: "John is a king and greedy."
- Conclusion: "John is evil."

Tip: It's like following a recipe. If the conditions are met, you get the result.

Unification in First-Order Logic (FOL)

Unification is a process used in **First-Order Logic** to make different logical expressions look the same by finding suitable substitutions. It is a key part of inference in FOL.

Key Idea:

- **Substitution**: Replacing variables with constants or terms to make two expressions identical.
- **UNIFY Algorithm**: Takes two logical expressions and finds a **unifier**—a set of substitutions that makes the expressions identical.

How Unification Works:

- Given two expressions **P(x, y)** and **P(a, f(z))**, you can substitute:
  - `x` with `a`
  - `y` with `f(z)`

  This makes the two expressions identical. The **substitution set** would be:
- `[a/x, f(z)/y]`

Example:

Given the predicate **Knows(John, x)** (Who does John know?), the **UNIFY** algorithm compares it with other sentences in the knowledge base:

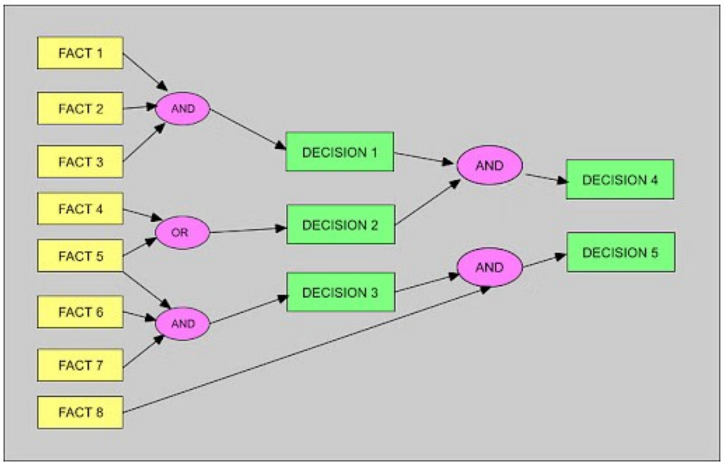
- **UNIFY(Knows(John, x), Knows(John, Jane))** → Substitution: `{x/Jane}`

Conditions for Unification:

1. **Same Predicate**: The predicate symbols must be identical.
2. **Same Number of Arguments**: Both expressions must have the same number of arguments.
3. **No Conflicting Variables**: Unification will fail if the same variable tries to take two different values.
   - Example: **UNIFY(Knows(John, x), Knows(x, Elizabeth))** will **fail** because `x` cannot be John and Elizabeth at the same time.

Forward Chaining:

Forward chaining, is a method involving inference or logical rules (facts) for data extraction. It is a bottom-up approach. It start with atomic sentences in the knowledge base and applies inference rules in the forward direction to extract more data until a goal is reached.



Let us say we have the following:

Fact 1: A dog is up for adoption through person A.

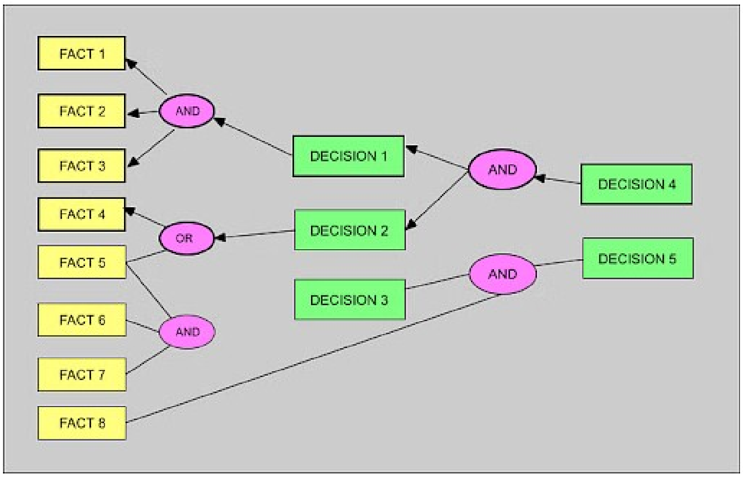
Fact 2: Person B is looking for a dog.

Inference rule: If a dog is up for adoption and someone is looking to adopt it, that person is free to adopt it.

Here, the decision can be reached as person b can adopt the dog from person A. This is how forward chaining works to make a decision.

Backward Chaining:

A backward chaining algorithm is a form of reasoning, which starts with the goal and works backward, chaining through rules to find known facts that support the goal. It is known as a top-down approach. The backward-chaining method mostly used a depth-first search strategy.



Example:

Decision/Goal: Person B adopts a dog.

Fact 1: A dog is up for adoption from Person A.

Fact 2: Person B is looking for a dog.

Inference rule: If a person wants to adopt a dog, he can if there is any up for adoption.

Here, the inference engine will begin with the goal and look if the conditions are met. If both conditions are met, the stated decision can be concluded.

Resolution in FOL:

Resolution is a method used in logic to prove something is true by showing that the opposite leads to a contradiction.

Key Idea: Assume the opposite of what you want to prove, then use **rules of logic** to find a contradiction.

Steps:

1. **Assume the opposite: Start by assuming the opposite of what you want to prove.**
   - **Goal**: Prove that Felix (Cat) is a mammal.
   - **Assume**: Felix is **not** a mammal (opposite of the goal).
2. **Convert sentences to clauses**: Break the facts into simple logical clauses.
   - Fact 1: x is mammal or x is not a cat → **Mammal(x)** ∨ **¬ Cat(x)**
   - Fact 2: Felix is a cat. → **Cat(Felix)**
3. **Unify the clauses**: Find matching parts to combine (in this case, "Cat(Felix)").
   - Match **Cat(Felix)** with **Mammal(x)** ∨ **¬ Cat(x)**
   - **Now we have Mammal(Felix) v ¬ Cat(Felix) (Meaning: Felix is a mammal or Felix is not a cat)**
4. **Resolve**: Eliminate the opposing terms to derive a conclusion.
   - Eliminate Opposites : cat(Felix) and **¬ Cat(Felix)** you're left with **Mammal(Felix)**.
5. **Contradiction**: You assumed Felix was not a mammal, but you've proven he **is** a mammal. The assumption was wrong, so the original statement is true!

Resolution shows that by assuming the opposite and reaching a contradiction, the original statement (Felix is a mammal) must be true.

Learning From Observation:

Learning involves changes that help a system perform tasks better over time. It's about improving efficiency and competence through experience.

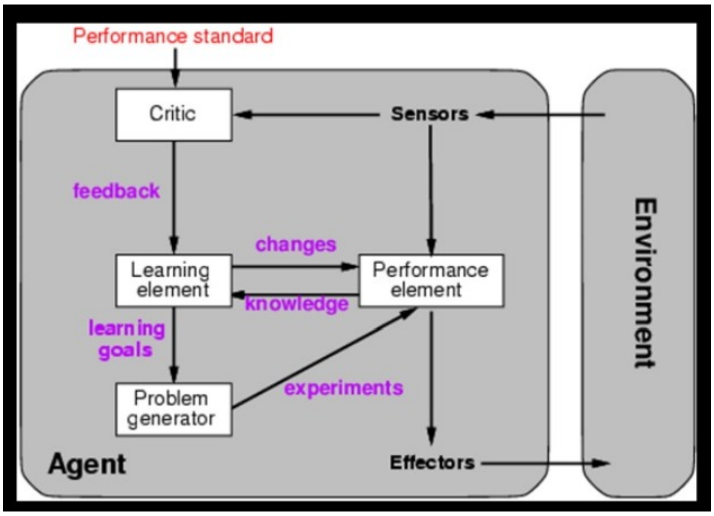
- **Key Idea**:
  - Learning means acquiring, organizing, or modifying knowledge.
  - Learning also involves creating new ideas or improving skills through practice.
  - It leads to positive changes in how well an agent (or person) performs tasks.

Why is Learning Important?

- Learning is essential in unknown environments where all details aren't known in advance.
- It helps systems adapt to reality and improve decision-making.

Forms of Learning

Learning Agents: A learning agent has four main components:



1. **Performance Element**: The part that decides the next action based on knowledge (e.g., walking, drawing).
2. **Learning Element**: This takes feedback from the critic and improves the performance element.
3. **Critic**: Provides feedback on how well the agent is performing based on standards (e.g., audience response).
4. **Problem Generator**: Suggests new actions for the agent to explore and improve.

Types of Learning:

- **Supervised Learning**: Learns from examples with correct answers.
- **Unsupervised Learning**: Learns patterns without knowing the correct answers.
- **Reinforcement Learning**: Learns by receiving occasional rewards for correct actions.

Inductive Learning:

Inductive learning is a method in artificial intelligence (AI) where the system learns from examples and observations. This allows the AI to make predictions or decisions about unseen data.

- **Learning from Examples**: The system is provided with examples (data points) and it tries to infer a general rule from these examples.
- **Generalization**: The system learns a general rule that applies not only to the examples it has seen but also to new, unseen situations.

Example of Inductive Learning:

If an AI sees several examples of birds that can fly (like sparrows, eagles, and crows), it may generalize the rule: "All birds can fly." However, the system may need to refine this rule when it encounters an exception (like penguins, which are birds but cannot fly).

How Inductive Learning Works:

1. **Training Data**: The AI is provided with a set of labeled data (examples with known outcomes).
2. **Pattern Recognition**: The AI identifies patterns or common features in the data.
3. **Generalization**: The system formulates a rule or model that works for the given data and applies it to new, unseen data.

Uses of Inductive Learning:

- **Classification**: Grouping items into categories (e.g., recognizing whether an email is spam or not).
- **Prediction**: Predicting future outcomes based on past data (e.g., predicting house prices based on features like size and location).

Inductive Learning Algorithms:

- **Decision Trees**: A model that splits data into branches based on conditions.
- **Neural Networks**: Systems that mimic the human brain to recognize patterns and make predictions.

Decision Tree:

A **Decision Tree** is a supervised learning algorithm used for classification and regression tasks.

- It resembles a tree structure, where:
    - **Internal nodes** represent decisions based on features.
    - **Branches** represent the outcome of those decisions.
    - **Leaf nodes** represent the final classification or result.

Terms Associated:

1. **Root Node**: The starting point of the tree; represents the entire dataset.
2. **Leaf Node**: The end node, representing the final decision (output).
3. **Decision Node**: A node where decisions are made, leading to branches.
4. **Splitting**: Dividing a node into sub-nodes based on feature conditions.
5. **Pruning**: Removing unnecessary branches to simplify the tree.
6. **Parent/Child Nodes**: Parent nodes are split into child nodes.

Applications:

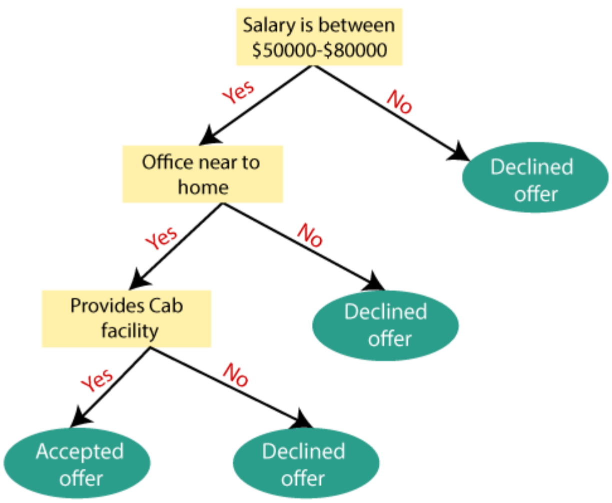
- **Human-like Decision Making**: They mimic the way humans make decisions, making them easy to understand.
- **Visual Representation**: The tree structure provides a clear view of decision paths.

Steps:

1. **Start** at the root node, which contains the entire dataset.
2. **Select the Best Feature**: Use **Attribute Selection Measures** (ASM) to find the best feature.
3. **Split** the dataset into subsets based on the best feature.
4. **Create Subtrees**: Continue splitting until you can't split further, reaching leaf nodes.
5. **Final Decision**: Once you reach a leaf node, you have your final classification.

Example:

A person decides whether to accept a job offer. The decision tree may start with the **Salary** as the root node, and split into further decisions like **Distance from the office**, then **Cab Facility**, and finally lead to leaf nodes: **Accept offer** or **Decline offer**.

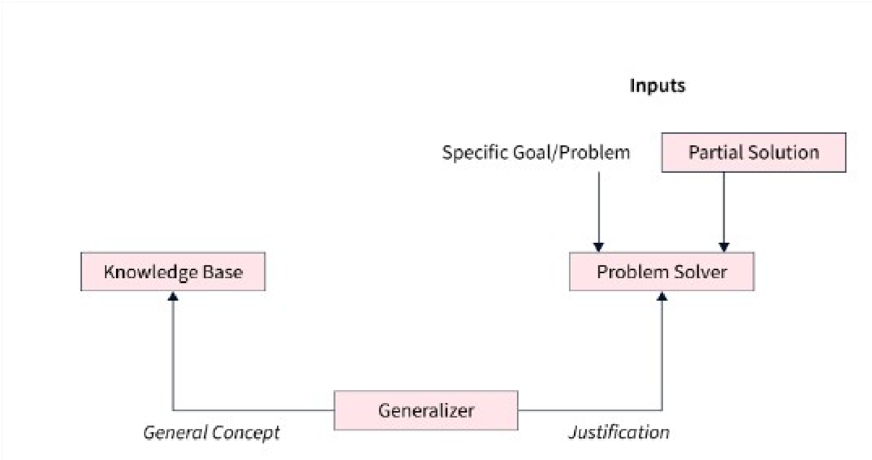


Explanation Based Learning:

A problem-solving approach in AI where learning happens by analyzing specific situations and applying prior knowledge to solve similar problems.

- EBL uses **logical reasoning** and **domain knowledge** rather than just statistical analysis.

EBL Architecture:



- **Inputs**:
    1. A specific goal.
    2. A partial solution.
- **Problem Solver**: Analyzes the inputs using training instances (examples) and inference rules (facts).
- **Generalizer**: Compares problem solver's output to the knowledge base and generalizes the solution.
- **Operational Pruner**: Refines the concept to match the desired output format.

EBL Hypothesis:

- Learning through explanations is more efficient than learning through examples alone. If a system can explain how it solved a similar problem before, it can apply that explanation to new problems more effectively.

Steps in EBL:

1. Identify the problem.
2. Analyze previously solved problems.
3. Find connections between old and new problems.
4. Extract rules and principles used in past solutions.
5. Apply those rules to the new problem.

Examples:

- **Medical Diagnosis**: Analyzing past diagnoses to make accurate predictions.
- **Robot Navigation**: Learning successful navigation strategies and applying them in new environments.

- **Fraud Detection**: Finding patterns of fraudulent behavior from past cases and applying them to detect new frauds.

Statistical Learning Methods:

• Statistical Learning based on the Learning of uncertainty in real environments.

• The methods probability and decision theory are used to handle uncertainty by the Agents

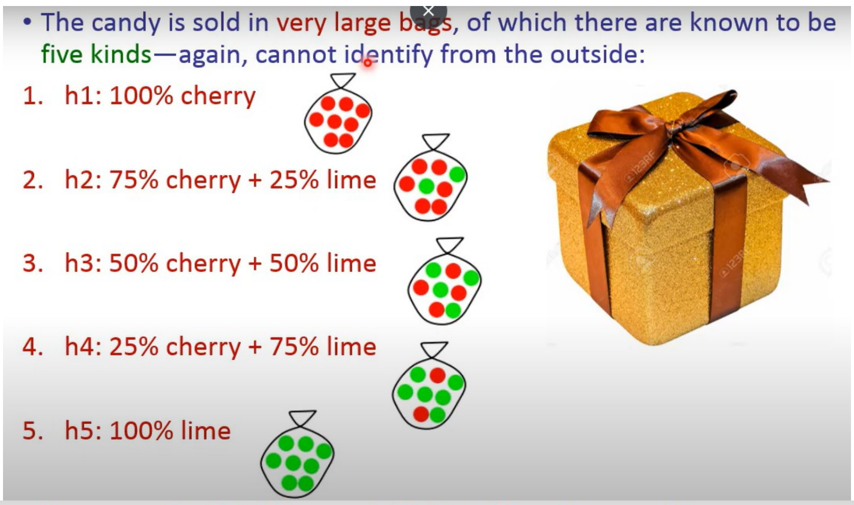
• First the agent must learn its probabilistic theories of the world from experience.

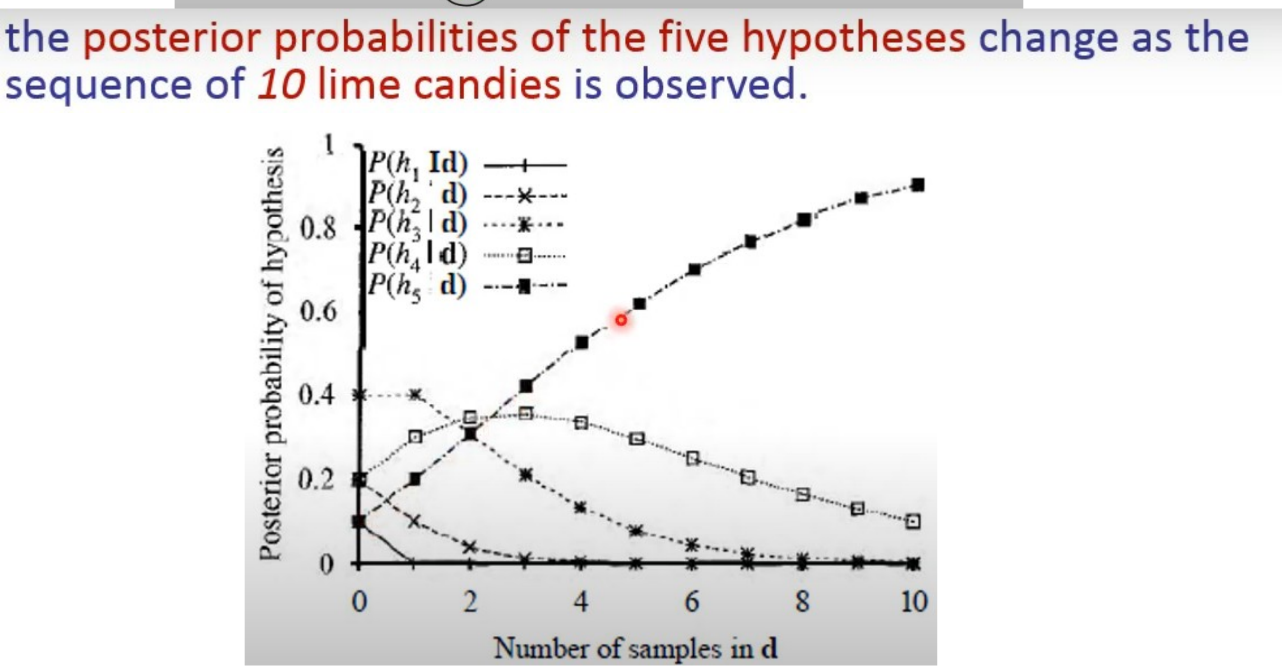
• A Bayesian view of learning is extremely powerful, providing general solutions.

• Statistical Learning is about inferences

• The idea is generated from the **Data and Hypothesis** and these are called as key terms of statistical learning.

• Data (Samples and Population) are Evidence





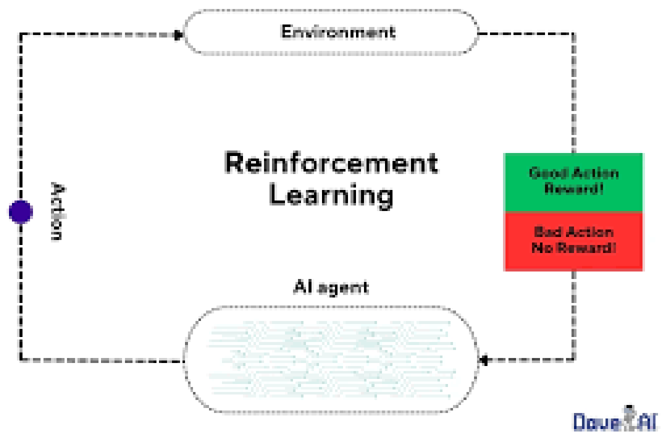
Reinforcement Learning:

A machine learning method that trains software (agents) to take actions in an environment based on rewards and punishments.

- Agents learn through **trial and error** to optimize their behavior for desired outcomes.

How RL Works:

- An agent perceives its environment and selects actions.
- Actions are met with feedback:
    - **Positive rewards** for desired actions.
    - **Negative penalties** for undesired actions.
- Over time, the agent learns to choose actions that maximize overall rewards.



Key Concepts:

- **Markov Decision Process**: A framework used in RL where an agent exists in a state and must choose the best action from possible options to receive rewards.
- **Cumulative Reward**: The total sum of rewards an agent accumulates from its actions over time.

Types of Reinforcement Learning Algorithms:

1. Model-based RL enables an agent to create an internal model of an environment. This lets the agent predict the reward of an action. The agent's algorithm is also based on maximizing award points. Model-based RL is ideal for static environments where the outcome of each action is well-defined.

2. Model-free RL uses a trial-and-error approach in an environment. The agent performs different actions multiple times to learn the outcomes. As it performs these actions, it creates a strategy -- called a policy -- that optimizes its reward points. Model-free RL is ideal for unknown, changing, large or complex environments.

Applications of Reinforcement Learning:

- **Gaming**: Achieves superhuman performance (e.g., in games like Pac-Man).
- **Robotics**: Teaches robots to learn tasks through interactions.
- **Resource Management**: Allocates resources efficiently to achieve specific goals.
- **Military**: Prepares autonomous vehicles for real-life scenarios and simulations.

Benefits of Reinforcement Learning:

- Works well in complex environments (both static and dynamic).
- Requires minimal human supervision.
- Focuses on long-term optimization and cumulative rewards.