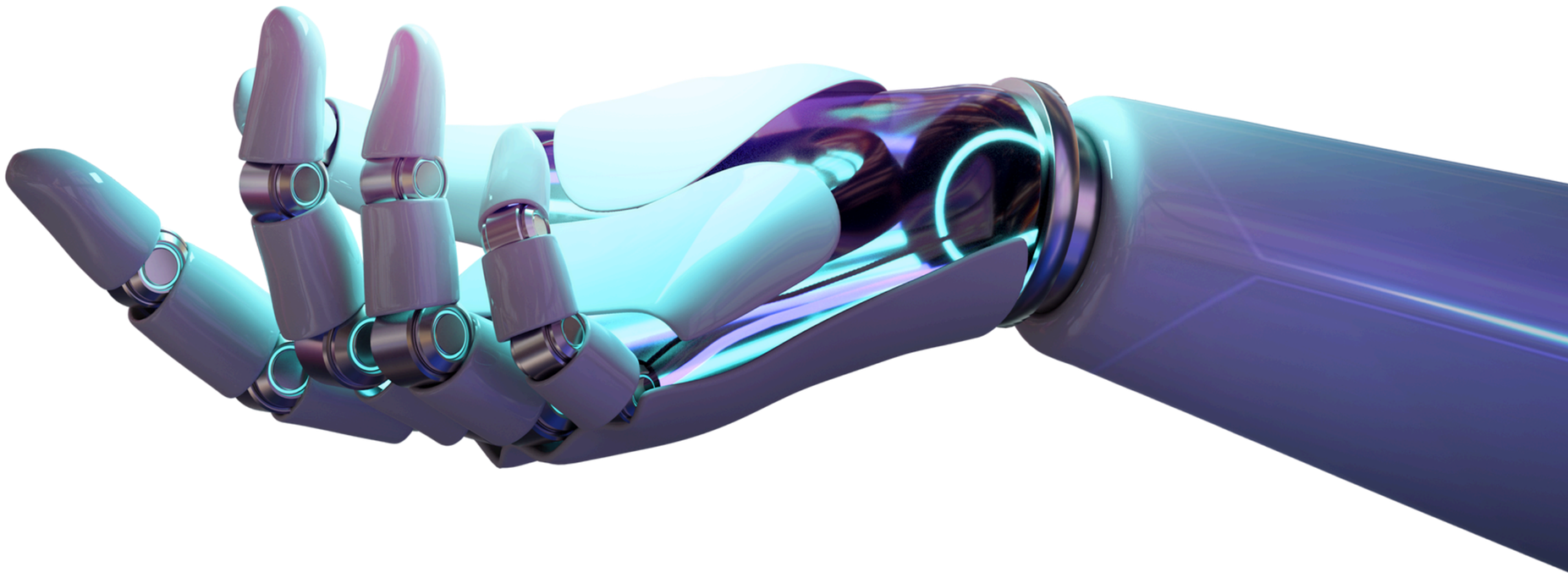
REINFORCEMENT LEARNING



Dr. Basel Magableh
basel.magableh@dbs.ie

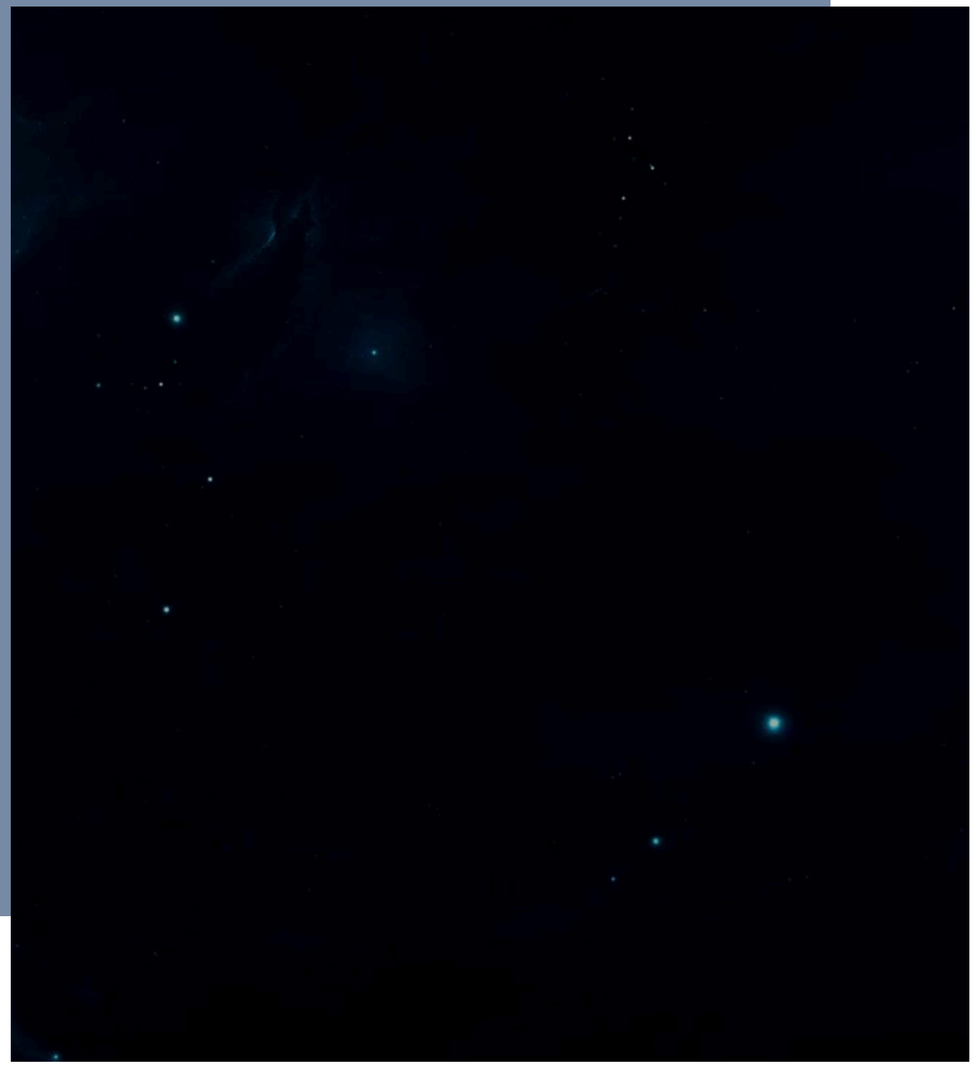
FUNDAMENTALS OF REINFORCEMENT LEARNING

Key elements of RL
The basic idea of RL
The RL algorithm

How RL differs from other ML paradigms.
The Markov Decision Processes.

Fundamental concepts of RL

Applications of RL



READING LISTS

SUTTON, R.S. AND BARTO, A.G. (2018). REINFORCEMENT LEARNING: AN INTRODUCTION. 2ND EDN. MIT PRESS

Sanghi, N. (2021) Deep reinforcement learning with python. New York: Apress.

METHOD OF ASSESSMENT

CA ONE (60%)

Final Written Examination

What is artificial intelligence?

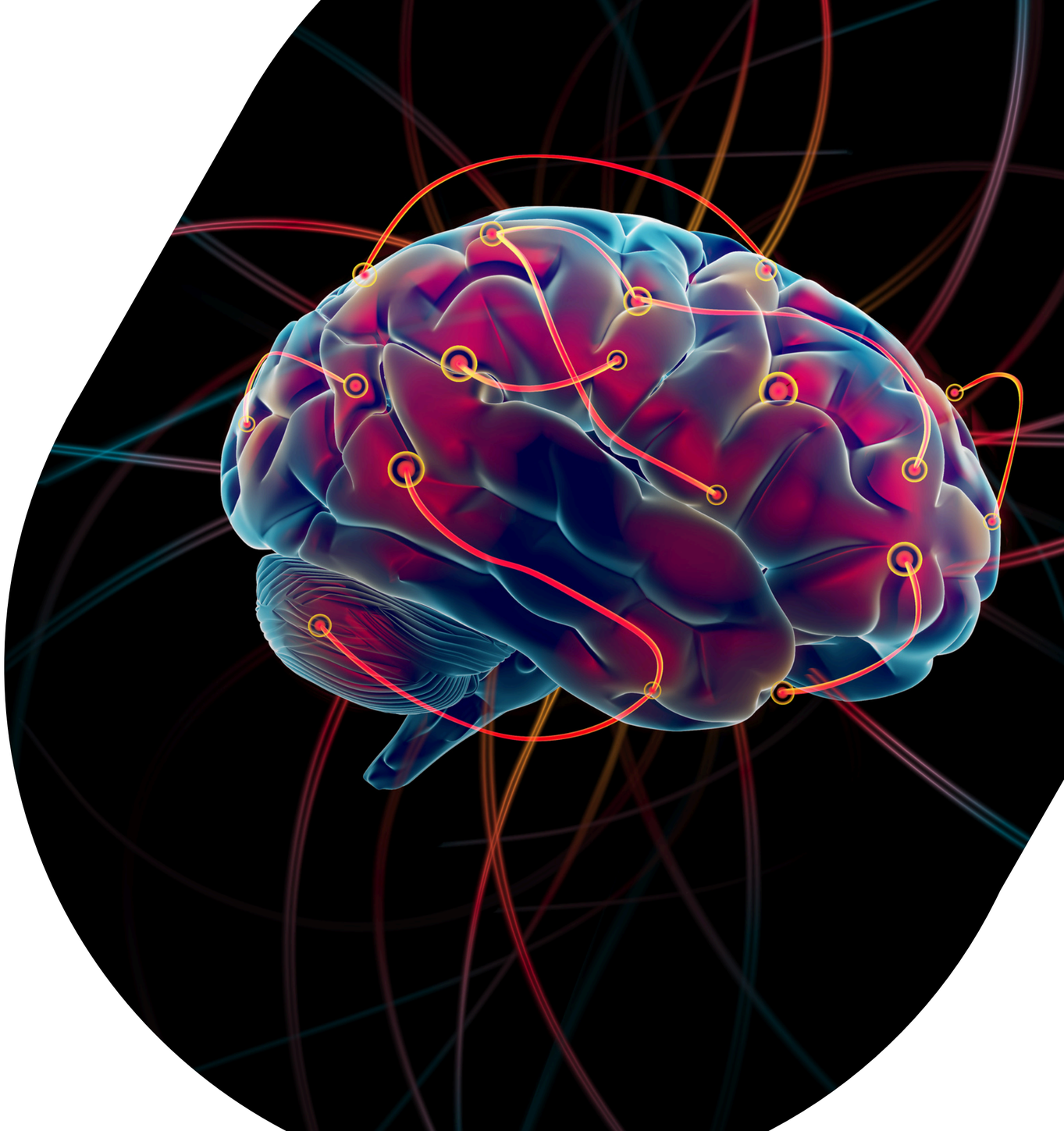
To be able to learn to make decisions to achieve goals

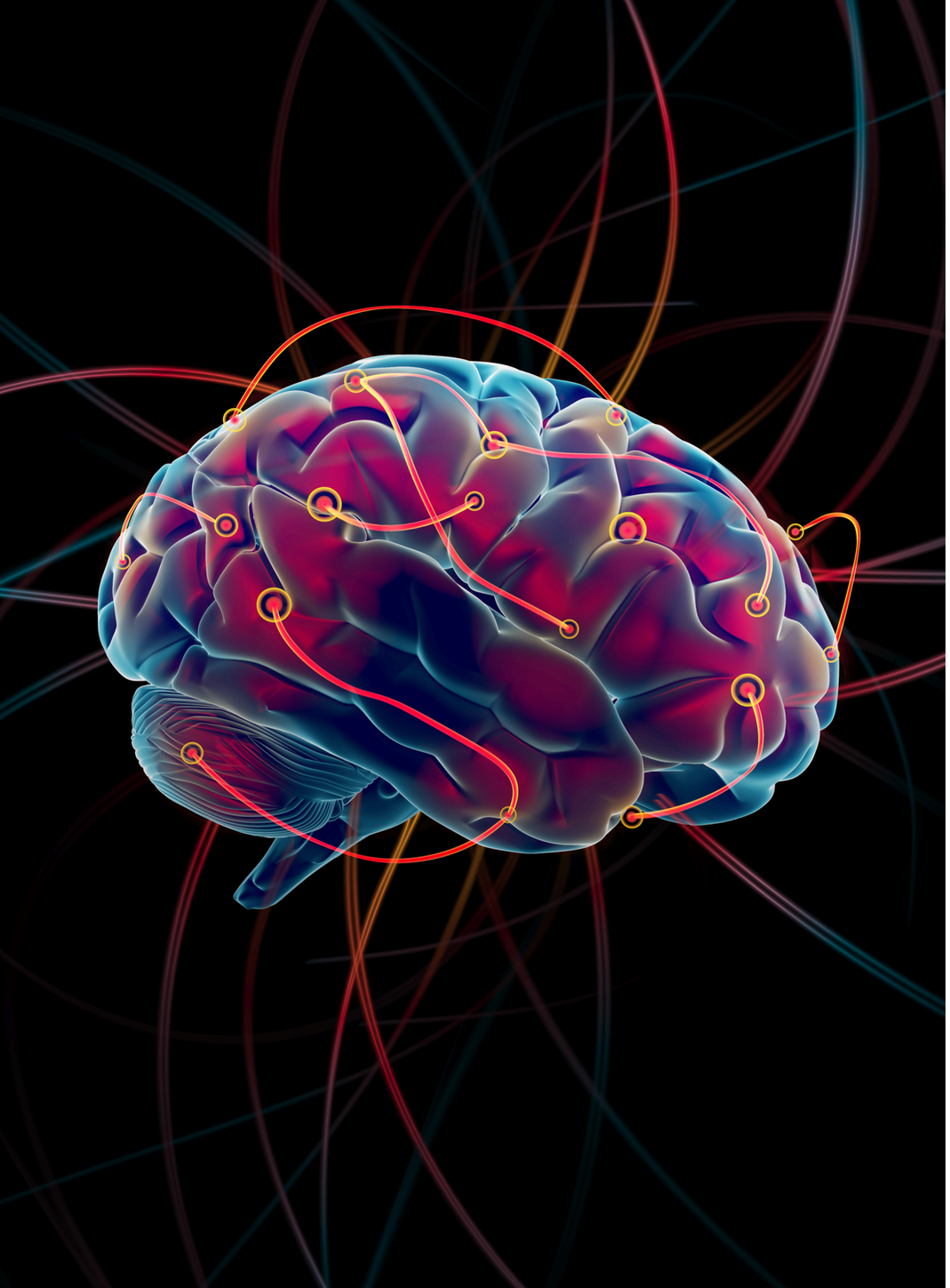
ARTIFICIAL INTELLIGENCE

Artificial Intelligence (AI) is a branch of computer science focused on developing intelligent machines that can perform tasks typically associated with human capabilities.

- AI aims to replicate human intelligence, learning and evolving as it progresses.
- While AI holds the potential to transform various industries, it also raises ethical and societal issues, including algorithmic fairness, transparency, accountability, and the effects of automation on employment and the economy.
- It is crucial to pursue AI development with care, empathy, and a human-centered approach.

WHAT IS REINFORCEMENT LEARNING?





Reinforcement Learning is a machine learning paradigm where agents learn optimal actions by interacting with an environment to maximize cumulative rewards.

WHAT IS REINFORCEMENT LEARNING?

Learning through Interaction:
How People and Animals Learn

Key Characteristics:

- Goal-directed learning through trial and error
- Feedback in the form of delayed rewards
- No need for labeled input/output pairs

REAL-WORLD EXAMPLES OF RL

01 Autonomous Vehicles: Learn to drive by minimizing collisions and following traffic rules

02 Robotics: Boston Dynamics robots learn to walk, run, or recover from a fall

03 Healthcare: Adaptive treatment strategies based on patient response

04 Finance: Portfolio optimization through market feedback

05 Gaming: AlphaGo beating human champions


SELF-DRIVING CARS LEARNING TO NAVIGATE TRAFFIC.
Robots Learning to Walk
FOR INSTANCE, BOSTON DYNAMICS SHOWCASES ADVANCEMENTS IN ROBOTICS.
Personalized Recommendations
TAKE NETFLIX, WHICH ADEPTLY LEARNS USER PREFERENCES TO SUGGEST RELEVANT SHOWS.

KEY CONCEPTS IN RL

Agent: Learner or decision-maker

Environment: Everything the agent interacts with

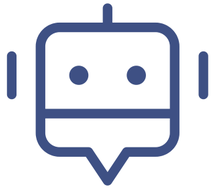
State: A snapshot of the environment at a moment

Action: What the agent chooses to do

Reward: Feedback signal to evaluate the action taken

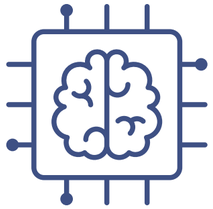
KEY ELEMENTS OF RL

AGENT



An agent is a software program or player that learns to make intelligent decisions, such as a chess player or a video game character like Mario. It is considered a learner in the Reinforcement Learning (RL) setting.

ENVIRONMENT



Defining the Environment in Agent-Based Systems
The environment is the space in which an agent operates, and remains within it. For instance, in a game of chess, the chessboard is considered the environment as the player (agent) learns and plays the game within that space. Similarly, in Super Mario Bros, Mario's world is known as the environment.

STATE AND ACTION

A state is a position or a moment in the environment that the agent can be in.

Understanding the Concept of States in an Environment for an Agent
It's important to learn that an agent remains within the environment and can occupy various positions in it. These positions are called states, and they're crucial to the performance of the agent. For example, in a game of chess, each position on the board is called a state. Typically, the state is represented by the letter "s".

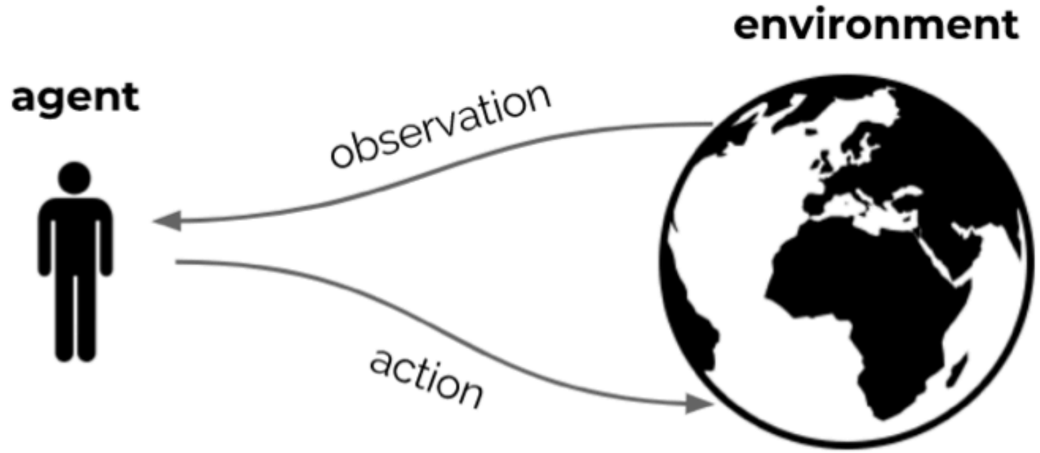
REWARD

The agent interacts with the environment, performs an action, and receives a reward based on the action, which is a numerical value. In chess, taking the opponent's piece is a good action resulting in a positive reward, while the opponent taking the agent's piece is a bad action resulting in a negative reward.

Understanding the Relationship Between the Agent and the Environment in Chess
In the game of chess, the agent interacts with the environment by making a move from one state to another. The agent, in this case, is the player. The move made by the player is referred to as an "action" and is often denoted by the letter "a".

INTERACTION LOOP

1. Agent observes the current state
2. Agent selects and performs an action
3. Environment transitions to a new state
4. Agent receives a reward
5. Agent updates strategy based on the reward



Goal: optimise sum of rewards, through repeated interaction

THE BASIC IDEA OF RL

In reinforcement learning, positive rewards are given to an agent for good actions and negative rewards for bad actions.

01

The agent starts with a random action, and if it is good, it receives a positive reward and repeats the action.

Imposing a Negative Penalty of -1 if the Robot Crashes into the Mountain and Gets Stuck.

02

If it is bad, the agent receives a negative reward and learns not to repeat that action.

When a robot walks in the correct direction without hitting a mountain, it receives a positive reward of +1.

THE RL ALGORITHM

- Agents try different actions in states
- Good actions are reinforced with positive rewards
- Bad actions are discouraged via penalties
- Over time, the agent learns a policy that maximizes rewards

The steps involved in a typical RL algorithm are as follows:

First, the agent interacts with the environment by performing an action.

By performing an action, the agent moves from one state to another.

Then the agent will receive a reward based on the action it performed.

Based on the reward, the agent will understand whether the action is good or bad.

If an agent receives a positive reward, it will prefer to perform that action. If not, it will try other actions to find a positive reward.

UNDERSTANDING THE ENVIRONMENT'S STATES AND AGENT'S GOAL

- The environment's states are referred to as positions A to I.
- The agent's objective is to go from state A to state I without stopping in the shaded states (B, C, G, and H).
- To achieve this goal, we will assign a negative reward (-1) whenever our agent visits a shaded state
- A positive reward (+1) when it visits an unshaded state.
- The agent can move up, down, right and left through the environment to reach state I from state A.

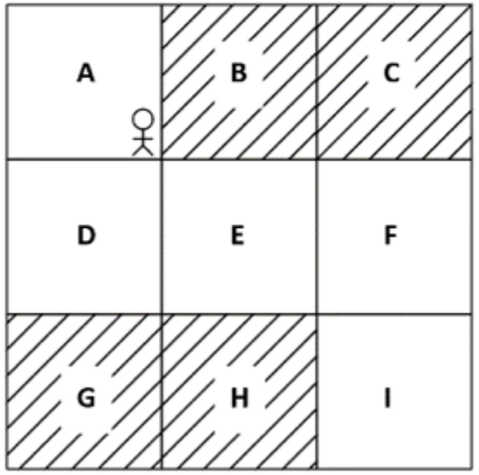


Figure 1.4: Grid world environment

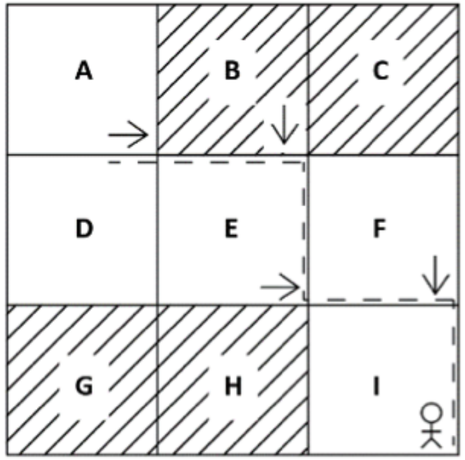


Figure 1.5: Actions taken by the agent in iteration 1

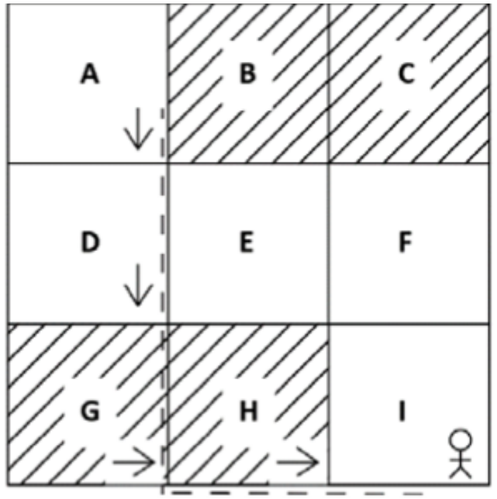


Figure 1.6: Actions taken by the agent in iteration 2

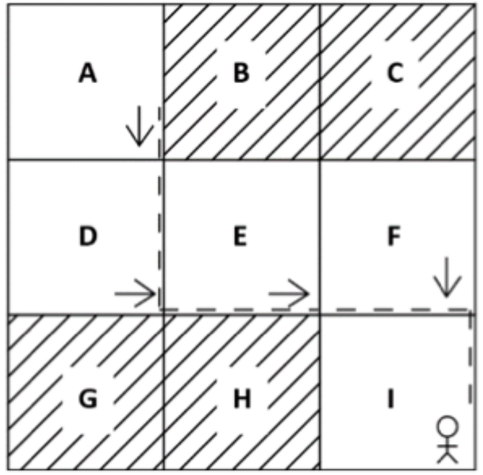
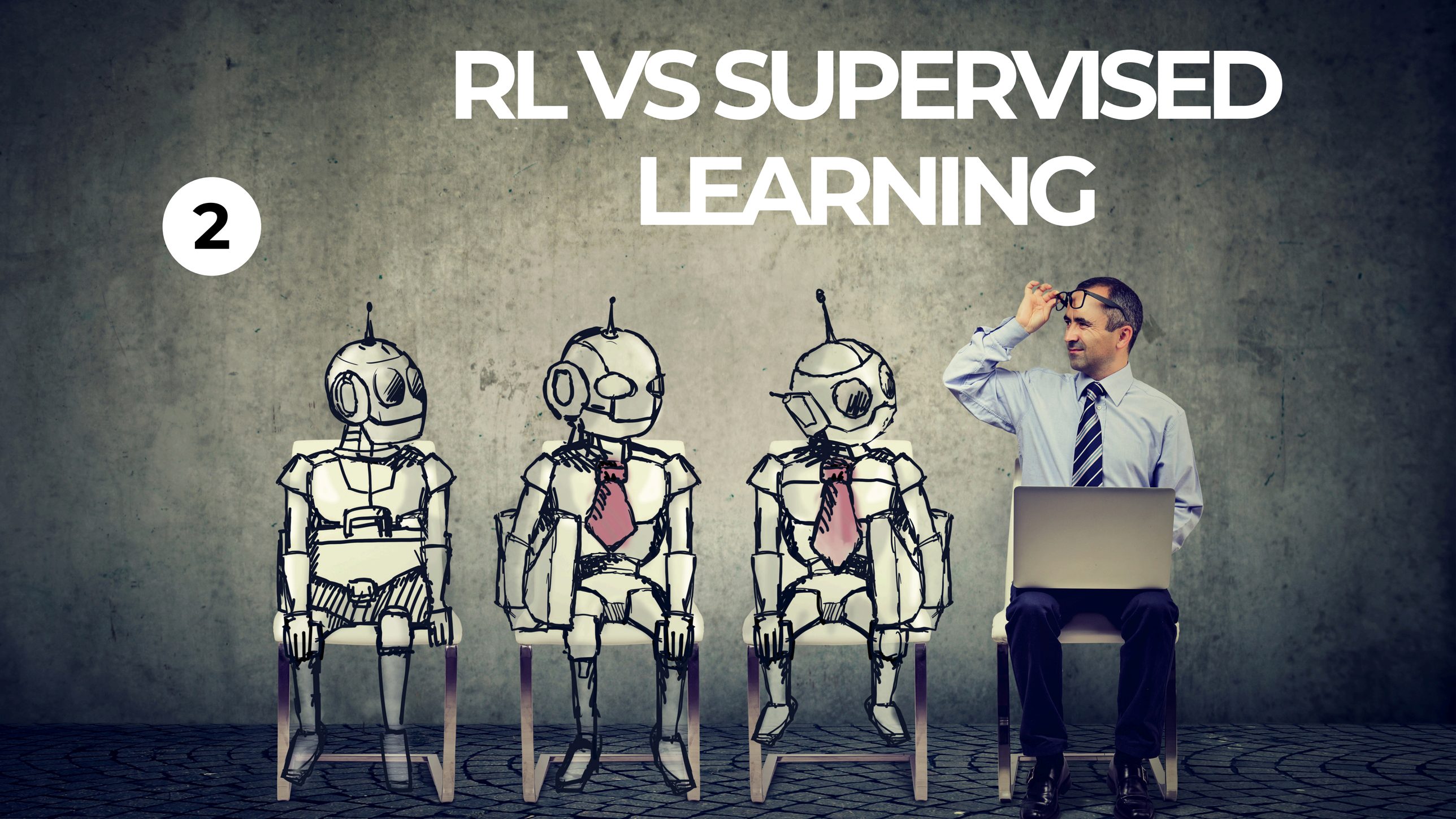


Figure 1.7: Actions taken by the agent in iteration 3

RL VS SUPERVISED LEARNING

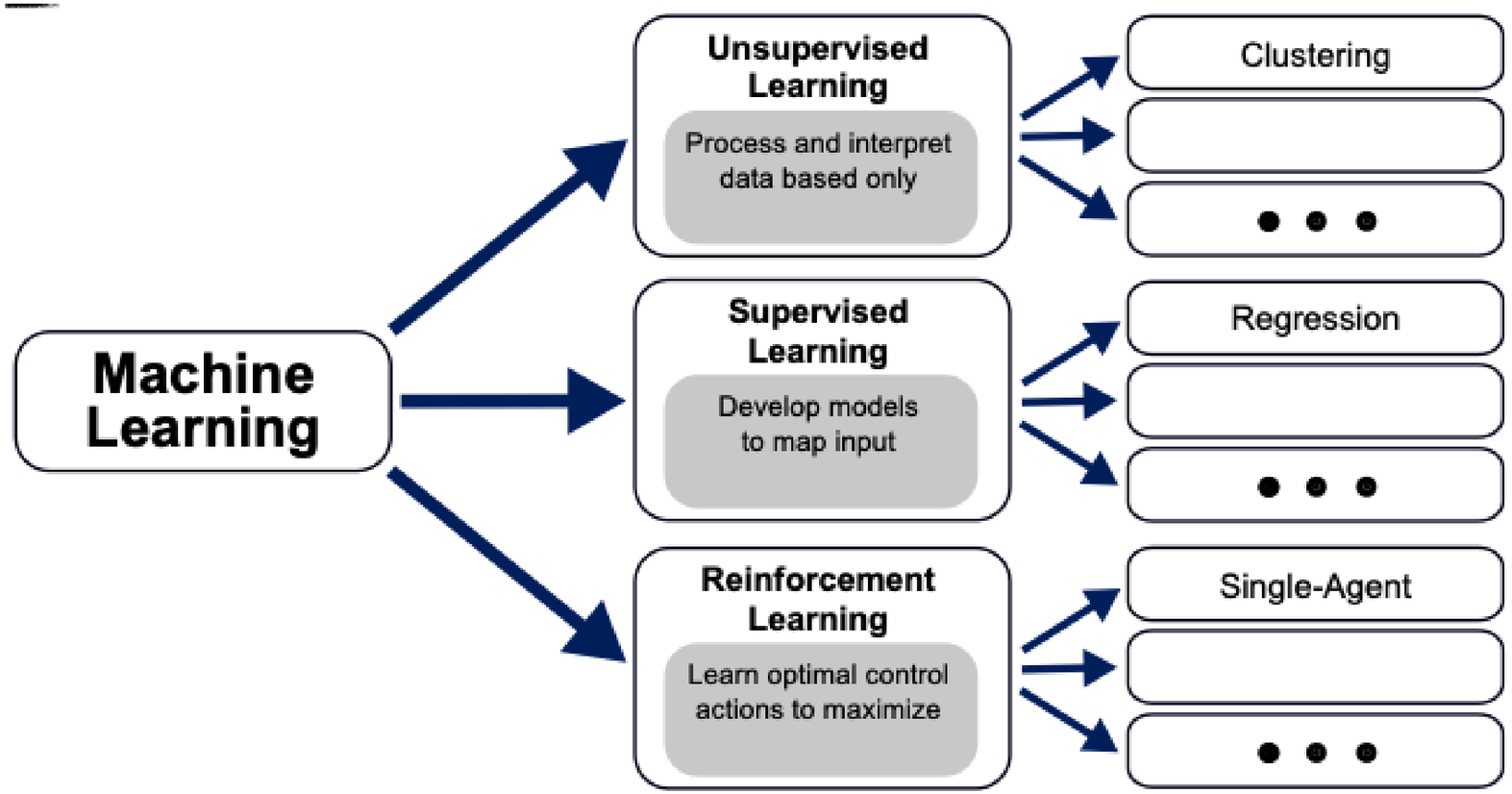
2



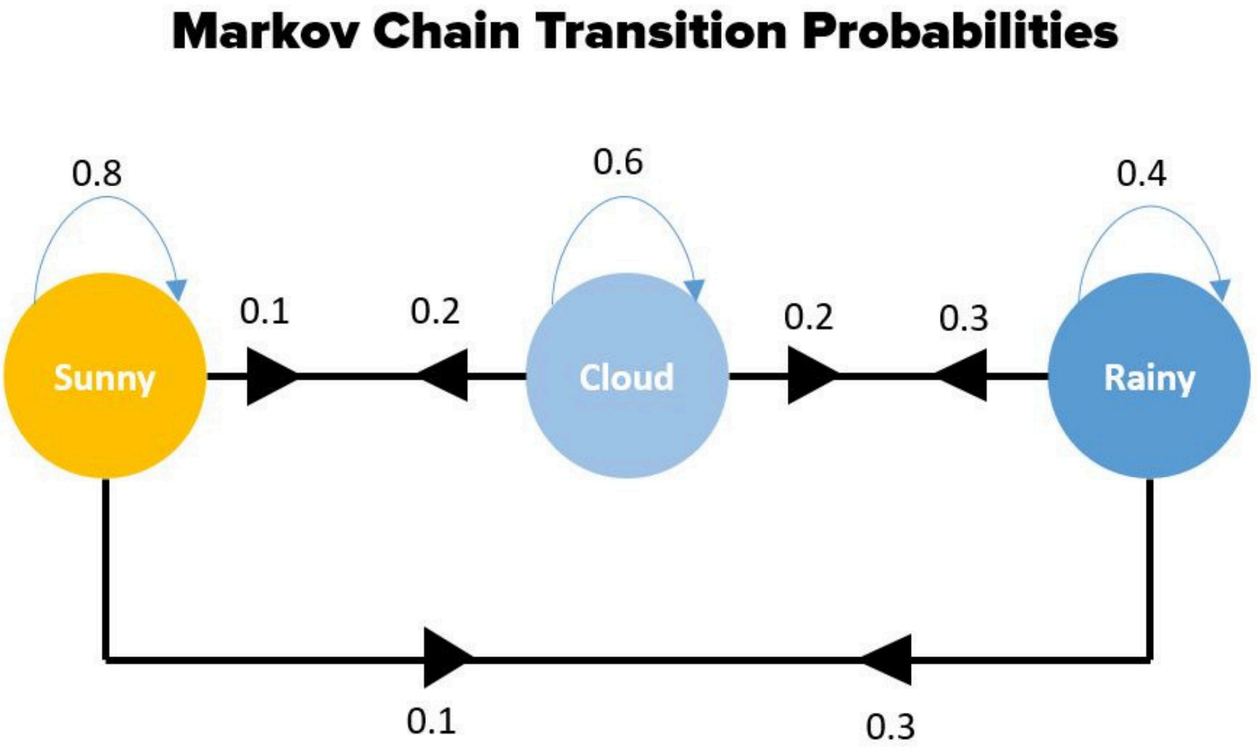
RL VS SUPERVISED LEARNING

Feature	Supervised Learning	Reinforcement Learning
Input	Labeled data	Environmental state
Output	Label prediction	Optimal action
Feedback	Immediate and correct	Delayed and evaluative
Example	Email spam filter	Robot navigation

HOW RL DIFFERS FROM OTHER ML PARADIGMS



MARKOV DECISION PROCESSES (MDP)



What is a Markov Chain?

Markov Chains are a class of Probabilistic Graphical Models (PGM) that represent dynamic processes i.e., a process which is not static but rather changes with time. In particular, it concerns more about how the ‘state’ of a process changes with time.

MARKOV DECISION PROCESSES (MDP)

Using the current state of weather, such as cloudy, can help predict the next state, such as rain, without considering the previous states.

- Moving from state to another is called a transition and it’s property called transition property.
- We denoate the transition property as P(S’|S)
- It indicates the property of moving from State S to the next state S’

MARKOV DECISION PROCESSES (MDP)

Markov Chains are a class of Probabilistic Graphical Models (PGM) that represent dynamic processes i.e., a process which is not static but rather changes with time. In particular, it concerns more about how the ‘state’ of a process changes with time.

- The Markov property is a fundamental concept that asserts that the future is determined solely by the present, rather than the past.
- A sequence of states that adheres to this property is known as the Markov chain or Markov process.
- This probabilistic model relies solely on the current state to forecast the next state, and not the previous states.
- As a result, the future is conditionally independent of the past, making the Markov chain a powerful tool for predicting future events.

MARKOV DECISION PROCESS (MDP)

- MDP is an extension of the MRP that includes actions. While the MRP consists of states, a transition probability, and a reward function, the MDP also includes actions.
- The Markov property dictates that the next state depends solely on the current state and not on the previous state. This principle is applicable to the RL environment as well.
- Agents in RL environments make decisions based only on the current state and not on past states.

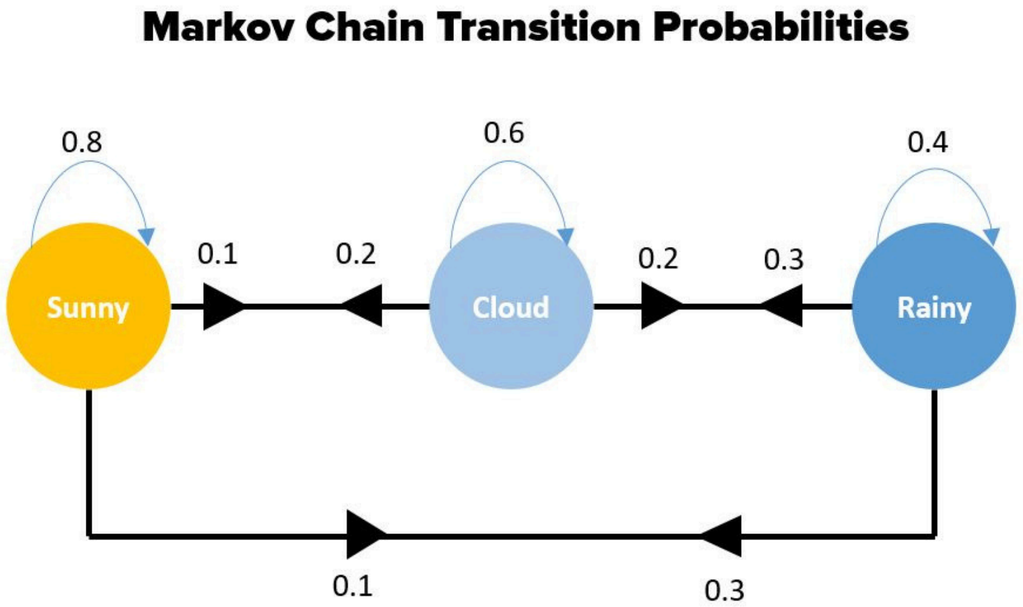
THEREFORE, WE CAN MODEL AN RL ENVIRONMENT AS AN MDP.

THE MARKOV REWARD PROCESS

- The Markov Reward Process (MRP) is an extension of the Markov chain with the reward function. That is, we learned that the Markov chain consists of states and a transition probability. The MRP consists of states, a transition probability, and also a reward function.
- A reward function tells us the reward we obtain in each state. For instance, based on our previous weather example, the reward function tells us the reward we obtain
- in the state cloudy, the reward we obtain in the state windy, and so on. The reward function is usually denoted by R(s).

The MRP consists of states s, a transition probability function $P(S'|S)$, and a reward function R(s).

MARKOV DECISION PROCESS (MDP)



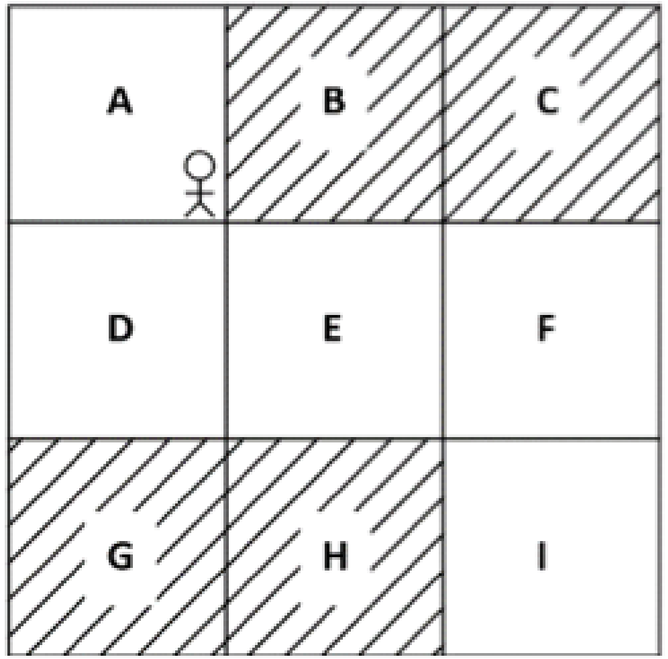
State	Action	Next Stat	Probabilit	Reward
Sunny	Wait	Sunny	0.8	10
Sunny	Wait	Cloud	0.1	2
Sunny	Wait	Rainy	0.1	-5
Cloud	Wait	Sunny	0.2	10
Cloud	Wait	Cloud	0.6	2
Cloud	Wait	Rainy	0.2	-5
Rainy	Wait	Sunny	0.3	10
Rainy	Wait	Cloud	0.3	2
Rainy	Wait	Rainy	0.4	-5

Lab 1: Introduction to Reinforcement Learning

MARKOV DECISION PROCESS EXAMPLE

In the Grid World Environment, the objective is for the agent to travel from state A to state I, without passing through any shaded states. The agent can only make decisions based on the current state and cannot refer to past states.

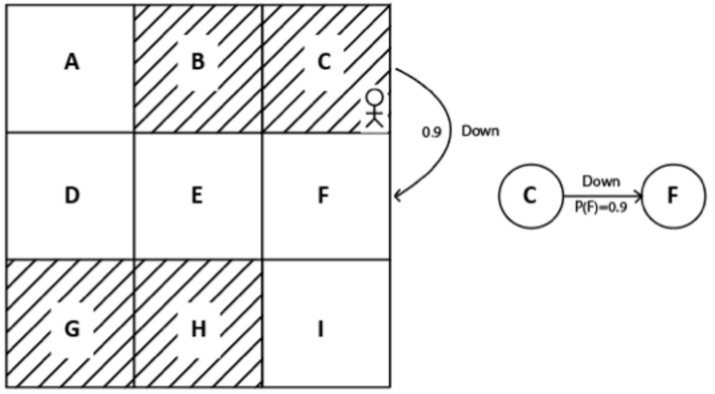
- States - The environment consists of a set of states, which are A through I in the grid world.
- Actions - The agent can choose to move up, down, left, or right, thereby transitioning to another state in the process.
- Transition probability – The transition probability is denoted by P(S'|S, a), implies the probability of moving from a state s to the next state ss' while performing an action a.



In our grid world environment, say the transition probability of moving from state A to state B while performing an action right is 100%. This can be expressed as P(B|A, right) = 1.0.

MARKOV DECISION PROCESS EXAMPLE

- Reward function - The reward function is denoted by R(s, a, s’) It re*ss*presents the reward our agent obtains while transitioning from state s to state s' while performing an action a.
- The reward we obtain while transitioning from state A to state B while performing the action right is -1, then it can be expressed as R(A, right, B) = -1
- Suppose our agent is in state C and say the reward we obtain while transitioning from state C to state F while performing the action down is +1, then it can be expressed as R(C, down, F) = +1



RL environment can be represented as an MDP with states, actions, transition probability, and the reward function.

Lab 2: Lunar Lander

Lab 3 : Frozen Lake

ACTION SPACE

DISCRETE ACTION SPACE

Discrete action space: When our action space consists of actions that are discrete, then it is called a discrete action space. For instance, in the grid world environment, our action space consists of four discrete actions, which are up, down, left, right, and so it is called a discrete action spa

CONTINUOUS ACTION SPACE

Continuous action space: When our action space consists of actions that are continuous, then it is called a continuous action space. For instance, let's suppose we are training an agent to drive a car, then our action space will consist of several actions that have continuous values, such as the speed at which we need to drive the car, the number of degrees we need to rotate the wheel, and so on. In cases where our action space consists of actions that are continuous, it is called a continuous action space.

POLICY

A policy defines the agent's behavior in an environment. The policy tells the agent what action to perform in each state. For instance, in the grid world environment, we have states A to I and four possible actions. The policy may tell the agent to move down in state A, move right in state D, and so on.

- When encountering a new environment, the first step towards learning is to create a random policy.
- This means the agent is instructed to perform a random action in each state, in order to determine the reward associated with each action.
- Through a series of iterations, the agent will learn to identify good actions in each state that lead to positive rewards.
- Ultimately, this process will lead to the development of a good policy that yields positive results.

This good policy is called the optimal policy. The optimal policy is the policy that gets the agent a good reward and helps the agent to achieve the goal.

POLICY

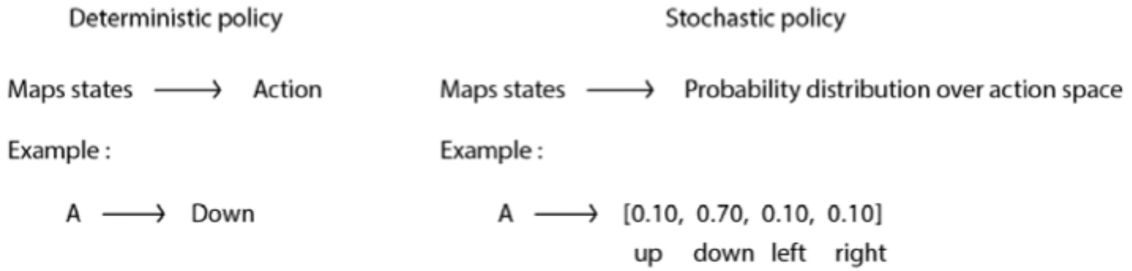
A deterministic policy directs an agent to take a specific action in a given state, mapping the state to one action. It is denoted by a function that specifies action 'a' for state 's' at time 't'.

$$a_t = \mu(s_t)$$

A deterministic policy

A stochastic policy maps a state to a probability distribution over actions, allowing an agent to perform different actions each time it visits the state, rather than always choosing the same action.

A stochastic policy



STOCHASTIC POLICY

CATEGORICAL POLICY

A stochastic policy is called a categorical policy when the action space is discrete.
That is, the stochastic policy uses a categorical probability distribution over the action space to select actions when the action space is discrete.

GAUSSIAN POLICY

A stochastic policy is called a Gaussian policy when our action space is continuous. That is, the stochastic policy uses a Gaussian probability distribution over the action space to select actions when the action space is continuous.

EPISODE

The agent interacts with the environment by performing some actions, starting from the initial state and reaches the final state.

This agent-environment interaction starting from the initial state until the final state is called an episode.

THE VALUE FUNCTION

The state value function, or value function, is a measurement of the value associated with a particular state. It represents the potential return an agent would receive when following policy π from that state.

Q FUNCTION

A Q function, also called the state-action value function, denotes the value of a state-action pair. The value of a state-action pair is the return the agent would obtain starting from state s and performing action a following policy. The value of a state- action pair or Q function is usually denoted by Q(s,a) and is known as the Q value or state-action value.

$$Q^{\pi}(s,a) = [R(\tau)|s_0 = s, a_0 = a]$$

4

Model-based and model-free learning

MODEL-BASED AND MODEL-FREE LEARNING

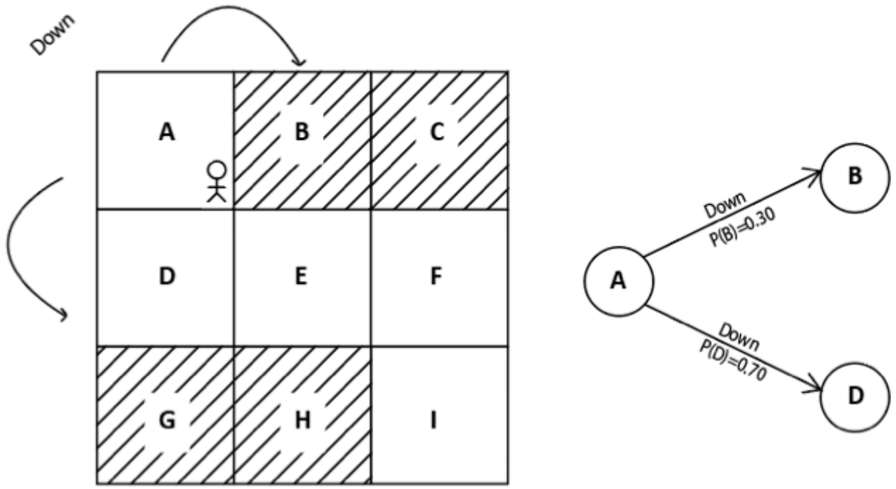
Model-based learning involves an agent having complete knowledge of the environment through a transition probability describing the likelihood of moving from one state to another and a reward function indicating the reward for taking a particular action. When the agent knows the transition probability, learning is called model-based learning. The agent can use the model dynamics to find the optimal policy.

Model-free learning: Model-free learning is when the agent does not know the model dynamics of its environment. That is, in model-free learning, an agent tries to find the optimal policy without the model dynamics.

DETERMINISTIC AND STOCHASTIC ENVIRONMENTS

Deterministic environment: In a deterministic environment, we are certain that when an agent performs action a in state s, then it always reaches state s' . For example, let's consider our grid world environment. Say the agent is in state A, and when it moves down from state A, it always reaches state D. Hence the environment is called a deterministic environment.

Stochastic environment: In a stochastic environment, we cannot say that by performing action a in state s the agent always reaches state s' because there will be some randomness associated with the stochastic environment.



DISCRETE AND CONTINUOUS ENVIRONMENTS

Discrete environment: A discrete environment is one where the environment's action space is discrete. For instance, in the grid world environment, we have a discrete action space, which consists of the actions [up, down, left, right] and thus our grid world environment is discrete.

Continuous environment: A continuous environment is one where the environment's action space is continuous. For instance, suppose we are training an agent to drive a car, then our action space will be continuous, with several continuous actions such as changing the car's speed, the number of degrees the agent needs to rotate the wheel, and so on. In such a case, our environment's action space is continuous.

EPISODIC AND NON-EPISODIC ENVIRONMENTS

Episodic environment: In an episodic environment, an agent's current action will not affect future actions, and thus an episodic environment is also called a non- sequential environment.

Non-episodic environment: In a non-episodic environment, an agent's current action will affect future actions, and thus a non-episodic environment is also called a sequential environment. For example, a chessboard is a sequential environment since the agent's current action will affect future actions in a chess match.

SINGLE AND MULTI-AGENT ENVIRONMENTS

Single and multi-agent environments

- Single-agent environment: When our environment consists of only a single agent, then it is called a single-agent environment.
- Multi-agent environment: When our environment consists of multiple agents, then it is called a multi-agent environment.

APPLICATIONS OF RL

Deep Reinforcement Learning (DRL) has driven significant growth in real-life applications, combining the strengths of RL and deep learning to solve various challenges. The most recent and cutting-edge DRL algorithms will be explored to expand understanding of this exciting technology.

SUMMARY

This chapter began by introducing the basic concept of Reinforcement Learning (RL), which is a trial and error learning process that relies on rewards. We then explored the distinctions between RL and other ML paradigms, such as supervised and unsupervised learning. We also delved into the Markov Decision Process (MDP) and how it can be used to model the RL environment. Throughout the chapter, several key RL concepts were discussed, and real-world applications of the technology were presented.

With these foundational RL concepts in mind, the next chapter will take a hands-on approach, using the popular toolkit known as Gym to implement everything learned thus far.

CHECK YOUR UNDERSTANDING

1. How does Reinforcement Learning vary from other Machine Learning paradigms?
2. What does the term 'environment' mean in the Reinforcement Learning context?
3. What distinguishes a deterministic policy from a stochastic policy?
4. What is an episode in Reinforcement Learning?
5. Why is the discount factor so essential?
6. How does the Q function differ from the value function?
7. What distinguishes a stochastic environment from a deterministic environment?

THANK YOU!

basel.magableh@icloud.com
if you have more questions.