Deep Learning

BCSE-332L

Module 7:

Deep Reinforcement Learning

Dr . Saurabh Agrawal

Faculty Id: 20165

School of Computer Science and Engineering

VIT, Vellore-632014

Tamil Nadu, India

Outline

- Deep Reinforcement Learning
- Q-Learning and Deep Q-Learning
- Policy Gradients
- Advantage Actor Critic (A2C)
- Asynchronous Advantage Actor Critic (A3C)
- Model based Reinforcement Learning
- Challenges

Deep Reinforcement Learning

- Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions.
- For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.
- In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning.
- Since there is no labeled data, so the agent is bound to learn by its experience only.
- RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive rewards.

Deep Reinforcement Learning

- The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way.
- ‘Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that."
- How a Robotic dog learns the movement of his arms is an example of Reinforcement learning.
- It is a core part of Artificial intelligence, and all AI agent works on the concept of reinforcement learning. Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention.

Deep Reinforcement Learning

❑**Example:** Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond.

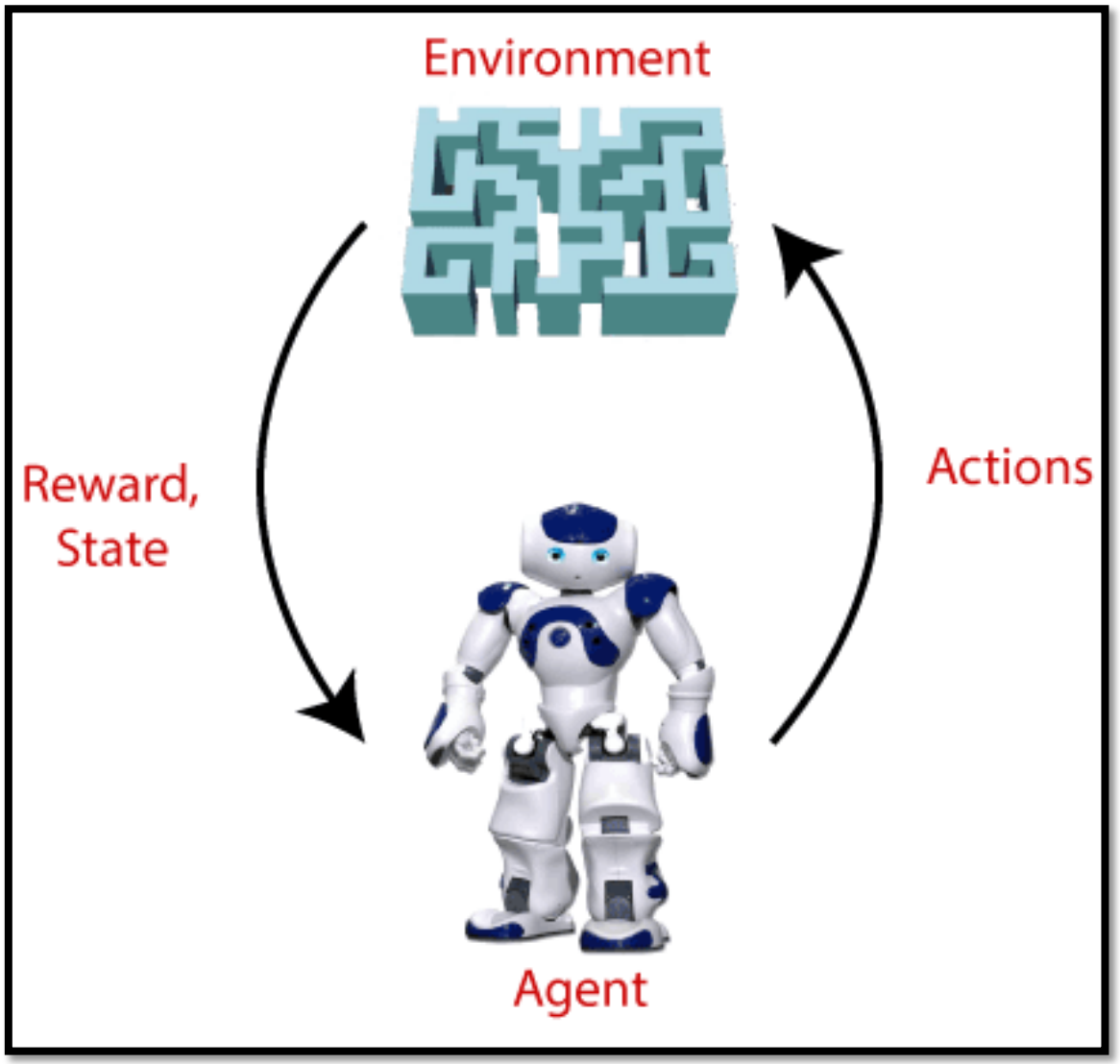
❑The agent interacts with the environment by performing some actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.

❑The agent continues doing these three things (**take action, change state/remain in the same state, and get feedback**), and by doing these actions, he learns and explores the environment.

❑The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty.

❑As a positive reward, the agent gets a positive point, and as a penalty, it gets a negative point.

Deep Reinforcement Learning



Deep Reinforcement Learning

❑**Example:** Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond.

❑The agent interacts with the environment by performing some actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.

❑The agent continues doing these three things (**take action, change state/remain in the same state, and get feedback**), and by doing these actions, he learns and explores the environment.

❑The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty.

❑As a positive reward, the agent gets a positive point, and as a penalty, it gets a negative point.

❑Terms used in Reinforcement Learning

1. **Agent():** An entity that can perceive/explore the environment and act upon it.
2. **Environment():** A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
3. **Action():** Actions are the moves taken by an agent within the environment.
4. **State():** State is a situation returned by the environment after each action taken by the agent.
5. **Reward():** A feedback returned to the agent from the environment to evaluate the action of the agent.
6. **Policy():** Policy is a strategy applied by the agent for the next action based on the current state.
7. **Value():** It is expected long-term retuned with the discount factor and opposite to the short-term reward.
8. **Q-value():** It is mostly similar to the value, but it takes one additional parameter as a current action (a).

❑Key Features of Reinforcement Learning

1. In RL, the agent is not instructed about the environment and what actions need to be taken.
2. It is based on the hit and trial process.
3. The agent takes the next action and changes states according to the feedback of the previous action.
4. The agent may get a delayed reward.
5. The environment is stochastic, and the agent needs to explore it to reach to get the maximum positive rewards.

Deep Reinforcement Learning

❑**Approaches to implement Reinforcement Learning:** There are mainly three ways to implement reinforcement-learning in ML, which are:

1. Value-based: The value-based approach is about to find the optimal value function, which is the maximum value at a state under any policy. Therefore, the agent expects the long-term return at any state(s) under policy π.

2. Policy-based: Policy-based approach is to find the optimal policy for the maximum future rewards without using the value function. In this approach, the agent tries to apply such a policy that the action performed in each step helps to maximize the future reward. The policy-based approach has mainly two types of policy:

- **Deterministic:** The same action is produced by the policy (π) at any state.
- **Stochastic:** In this policy, probability determines the produced action.

3. Model-based: In the model-based approach, a virtual model is created for the environment, and the agent explores that environment to learn it. There is no particular solution or algorithm for this approach because the model representation is different for each environment.

Deep Reinforcement Learning

❑**Elements of Reinforcement Learning:** There are four main elements of Reinforcement Learning, which are given below:

1. Policy
2. Reward Signal
3. Value Function
4. Model of the environment

❑Elements of Reinforcement Learning:

1) Policy: A policy can be defined as a way how an agent behaves at a given time. It maps the perceived states of the environment to the actions taken on those states. A policy is the core element of the RL as it alone can define the behavior of the agent. In some cases, it may be a simple function or a lookup table, whereas, for other cases, it may involve general computation as a search process. It could be deterministic or a stochastic policy

For deterministic policy: $a = \pi(s)$

For stochastic policy: $\pi(a \mid s) = P[A_t = a \mid S_t = s]$

❑Elements of Reinforcement Learning:

2) Reward Signal: The goal of reinforcement learning is defined by the reward signal. At each state, the environment sends an immediate signal to the learning agent, and this signal is known as a **reward signal**. These rewards are given according to the good and bad actions taken by the agent. The agent's main objective is to maximize the total number of rewards for good actions. The reward signal can change the policy, such as if an action selected by the agent leads to low reward, then the policy may change to select other actions in the future.

3) Value Function: The value function gives information about how good the situation and action are and how much reward an agent can expect. A reward indicates the **immediate signal for each good and bad action**, whereas a value function specifies **the good state and action for the future**. The value function depends on the reward as, without reward, there could be no value. The goal of estimating values is to achieve more rewards.

❑Elements of Reinforcement Learning:

4) Model: The last element of reinforcement learning is the model, which mimics the behavior of the environment. With the help of the model, one can make inferences about how the environment will behave. Such as, if a state and an action are given, then a model can predict the next state and reward.

The model is used for planning, which means it provides a way to take a course of action by considering all future situations before actually experiencing those situations. The approaches for solving the RL problems **with the help of the model** are termed as the **model-based approach**. Comparatively, an approach **without using a model** is called a **model-free approach**.

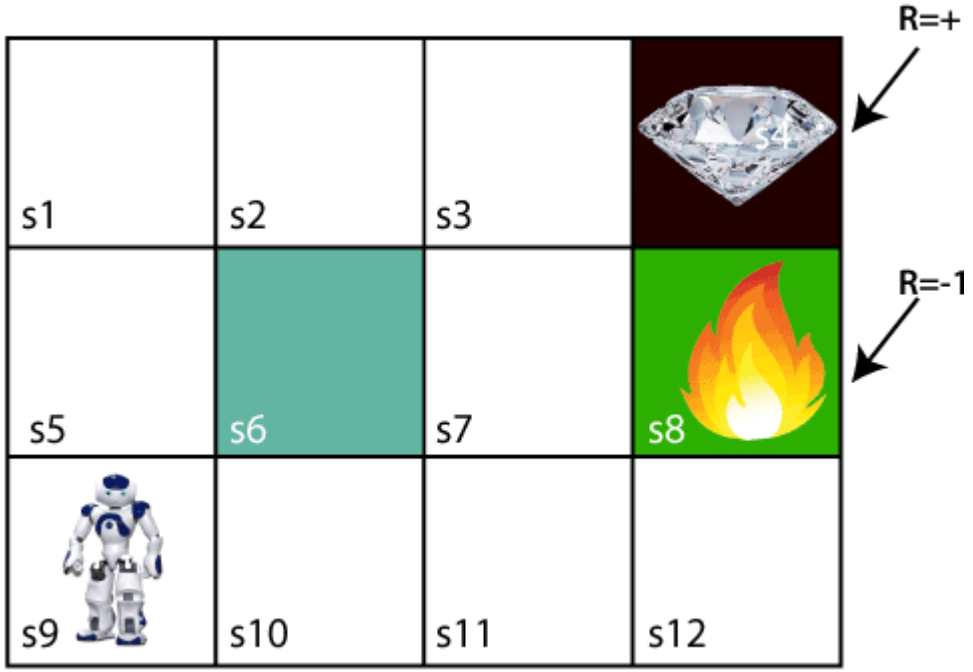
❑How does Reinforcement Learning Work?

❑To understand the working process of the RL, we need to consider two main things:

❑**Environment:** It can be anything such as a room, maze, football ground, etc.

❑**Agent:** An intelligent agent such as AI robot.

❑Let's take an example of a maze environment that the agent needs to explore. Consider the below image:



Lorem ipsum

❑How does Reinforcement Learning Work?

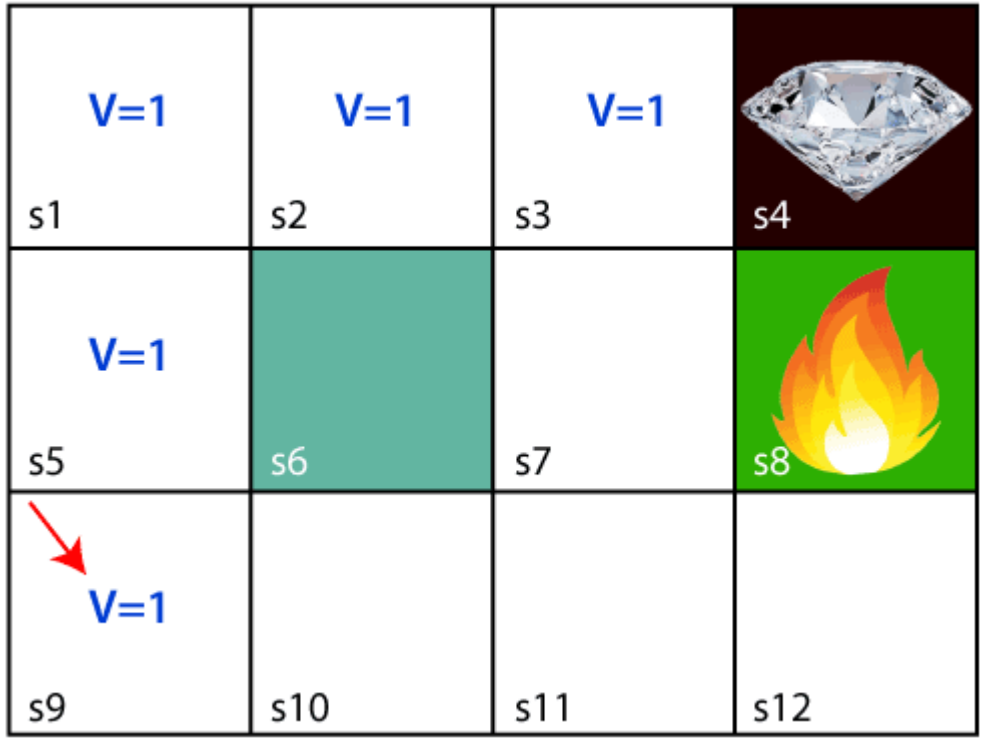
❑In the above image, the agent is at the very first block of the maze. The maze is consisting of an S_6 block, which is a **wall**, S_8 a **fire pit**, and S_4 a **diamond block.**

❑The agent cannot cross the S_6 block, as it is a solid wall. If the agent reaches the S_4 block, then get the **+1 reward;** if it reaches the fire pit, then gets **-1 reward point**. It can take four actions**: move up, move down, move left, and move right.**

❑The agent can take any path to reach to the final point, but he needs to make it in possible fewer steps. Suppose the agent considers the path **S9-S5-S1-S2-S3**, so he will get the +1-reward point.

❑The agent will try to remember the preceding steps that it has taken to reach the final step. To memorize the steps, it assigns 1 value to each previous step. Consider the below step:

❑How does Reinforcement Learning Work?

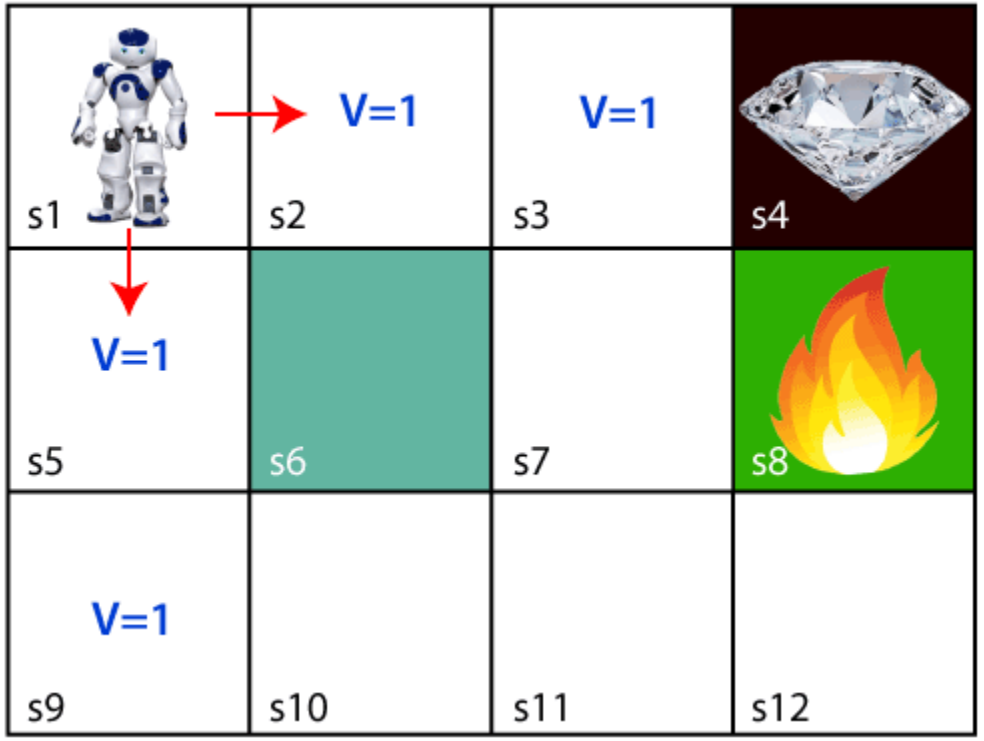


❑How does Reinforcement Learning Work?

❑Now, the agent has successfully stored the previous steps assigning the 1 value to each previous block.

❑But what will the agent do if he starts moving from the block, which has 1 value block on both sides?

❑Consider the below diagram:



❑Types of Reinforcement learning:

1. **Positive Reinforcement:** The positive reinforcement learning means adding something to increase the tendency that expected behavior would occur again. It impacts positively on the behavior of the agent and increases the strength of the behavior. This type of reinforcement can sustain the changes for a long time, but too much positive reinforcement may lead to an overload of states that can reduce the consequences.
2. **Negative Reinforcement:** The negative reinforcement learning is opposite to the positive reinforcement as it increases the tendency that the specific behavior will occur again by avoiding the negative condition. It can be more effective than the positive reinforcement depending on situation and behavior, but it provides reinforcement only to meet minimum behavior.

❑How to represent the agent state?

❑We can represent the agent state using the **Markov State** that contains all the required information from the history.

❑The State St is Markov state if it follows the given condition:

❑$P[S_t+1 \mid S_t] = P[S_t+1 \mid S_1, ..., S_t]$

❑The Markov state follows the **Markov property**, which says that the future is independent of the past and can only be defined with the present.

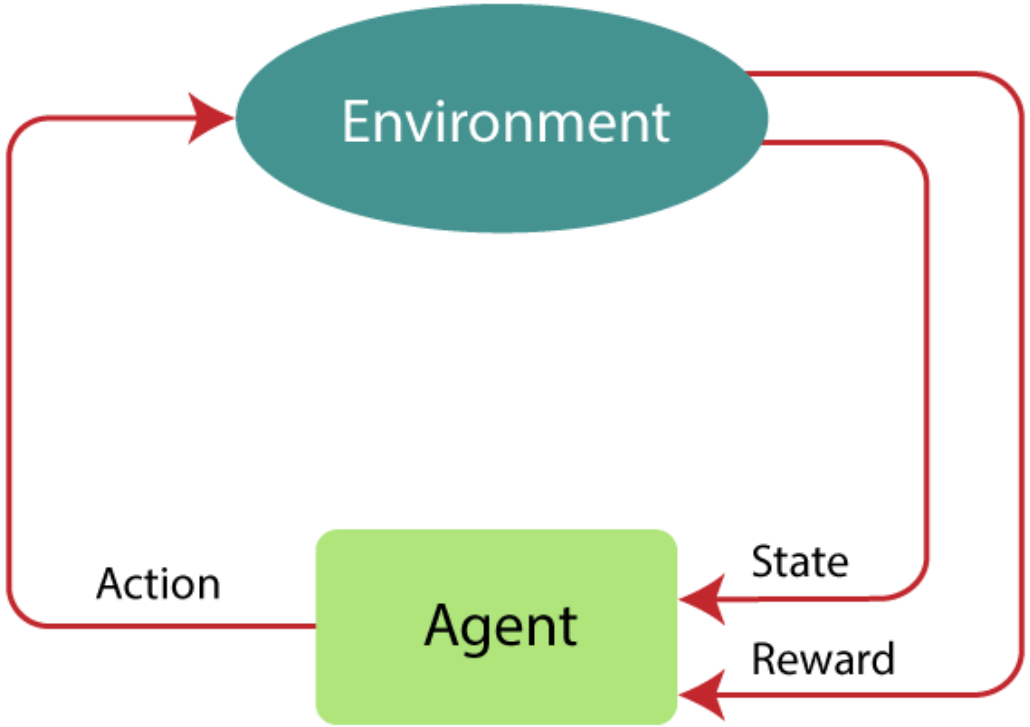
❑The RL works on fully observable environments, where the agent can observe the environment and act for the new state.

❑Markov Decision Process

❑Markov Decision Process or MDP, is used to **formalize the reinforcement learning problems**.

❑If the environment is completely observable, then its dynamic can be modeled as a **Markov Process**.

❑In MDP, the agent constantly interacts with the environment and performs actions; at each action, the environment responds and generates a new state.



❑Markov Decision Process

- ❑MDP is used to describe the environment for the RL, and almost all the RL problem can be formalized using MDP.
- ❑MDP contains a tuple of four elements (S, A, P_a, R_a):
- ❑A set of finite States S
- ❑A set of finite Actions A
- ❑Rewards received after transitioning from state S to state S', due to action a.
- ❑Probability P_a.
- ❑MDP uses **Markov property**, and to better understand the MDP, we need to learn about it.

Q-Learning and Deep Q-Learning

- Q-learning is a reinforcement learning policy that determines the next possible best action based on a current state.
- By choosing this action randomly, it strives to maximize its reward.
- The Q-learning algorithm is model-free, off-policy reinforcement learning, which finds appropriate action according to the agent's current state.
- Based on the agent's position within the environment, it decides what the next action will be.
- According to the model's current state, its goal is to determine the most appropriate course of action.
- To accomplish this, it may develop its own guidelines or not follow the pre-defined guidelines.
- As a result, no policy is actually necessary, which is why it is called off-policy.
- The environment tends to respond as expected when an agent uses model-free behavior.
- It prefers the trial and error method rather than the reward and punishment system.

Q-Learning and Deep Q-Learning

- The advertisement recommendation system is an excellent example of Q-learning.
- The normal ad recommendation system works as ads appear according to past browsing history or previous purchases.
- For instance, after you buy a mobile phone, you'll receive recommendations for new models from different brands.

Q-Learning and Deep Q-Learning

❑Important terms in Q-learning

1. **States:** The State S indicates a particular agent's position in a given environment.
2. **Action:** Action A describes what the agent takes when it is in a given state.
3. **Rewards:** Agents will receive positive or negative rewards based on their actions.
4. **Episodes:** Agents can't perform an action after reaching a terminating state.
5. **Q-values:** A measure of the quality of an action, A, performed at a particular state, S. Q (A, S).
6. **Temporal difference:** The Q-value is calculated based on the values of the present state, the previous state, and the action.

Q-Learning and Deep Q-Learning

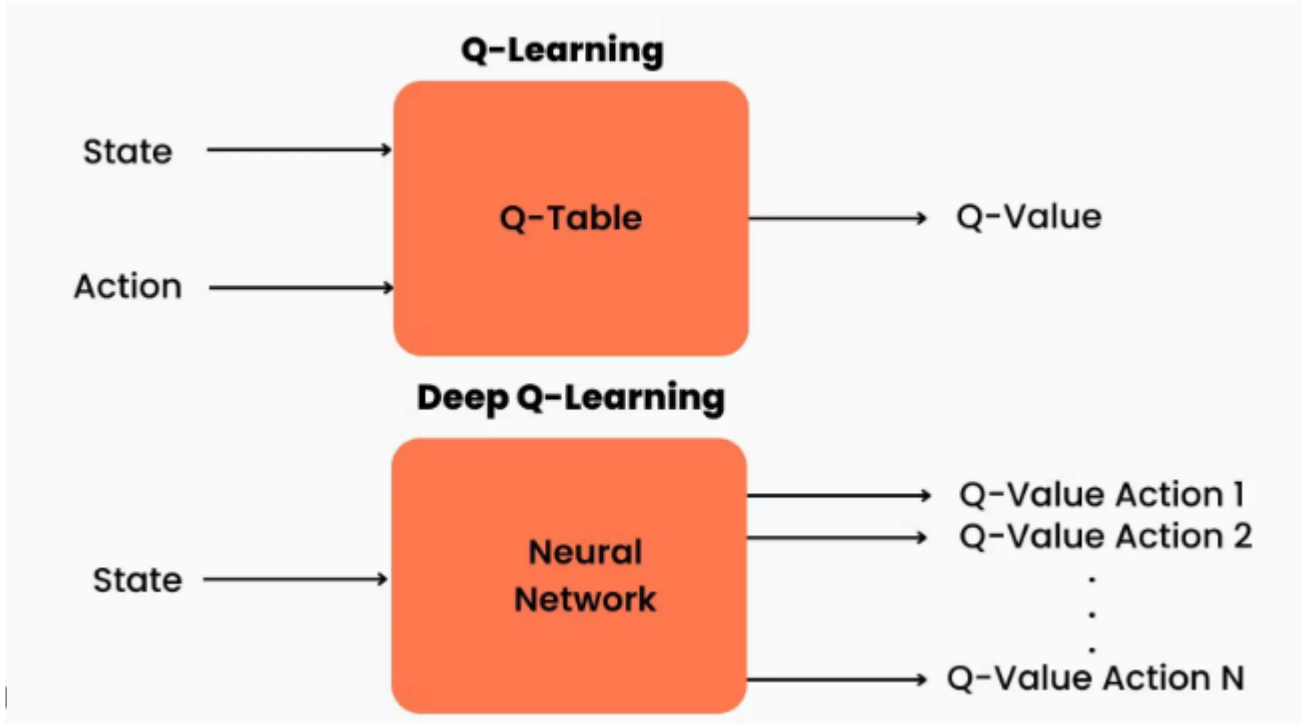
❑What is deep Q-learning?

❑The deep Q-learning model breaks the chain in order to find the optimal Q-value function.

❑It determines this by combining Q-learning and a neural network.

❑The uses of the deep Q-learning algorithm can be stated as finding the input and the optimal Q-value for all possible actions as the output.

❑The following image illustrates the differences between Q-learning and deep Q-learning:



Q-Learning and Deep Q-Learning

❑What is deep Q-learning?

❑In deep Q-learning, past experiences are stored in memory and the future action depends on the Q-network output.

❑It is how Q-network calculates the Q-value at state st. Similarly, the target network (neural network) determines the Q-value for state St+1 (next state) to stabilize the training.

❑As an additional feature, it copies Q-value count as training dataset for each iteration of Q-value in the Q-network, thereby blocking abrupt increases in Q-value count.

❑The deep Q-learning algorithm relies on neural networks and Q-learning. In this case, the neural network stores experience as a tuple in its memory with a tuple that includes <State, Next State, Action, Reward>.

❑A random sample of previous data increases the stability of neural network training.

❑As a result, deep Q-learning uses another concept to boost agents' performance – experience replay or store experience from the past.

Q-Learning and Deep Q-Learning

❑What is deep Q-learning?

❑The target network uses experience replay to determine the Q-value while the Q-network uses it for training.

❑Calculating the loss becomes easier when the squared difference between the target and predicted Q-values is known.

❑The equation is given below:

$$Loss = (r + \gamma max_{a'} Q(s', a') - Q(s, a))^2$$

Q-Learning and Deep Q-Learning

❑What is deep Q-learning?

❑To understand deep Q-learning better, we need to break it down into these steps:

1. First, the agent is informed about the environmental state.
2. The agent uses Q-values of possible actions based on the provided state.
3. The agent applies a Q-value to the action resulting in higher rewards and lower punishment.
4. We then follow up on rewards and next steps.
5. The agent keeps track of previous experiences from the action in an experience replay memory.
6. The agent uses experience replay memory to train the networks.

❑We continually repeat steps 2 through 6 according to the agent state.

Q-Learning and Deep Q-Learning

❑Why ‘deep’ Q-learning?

- ❑The Q-learning algorithm creates a cheat sheet for agents in a simple but quite powerful manner.
- ❑By doing so, the agents can determine exactly what action to take.
- ❑Consider an environment with 10,000 states and 1,000 actions per state. Wouldn't the cheat sheet be too long?
- ❑There would be a table with 10 million cells. Eventually, things would spiral out of control.
- ❑Thus, it would be impossible to estimate the Q-value for new states based on previously explored states.
- ❑Two main problems would arise:

1. As the number of states increases, the amount of memory required to save and update the table would expand.
2. It would become impossible to explore every state to create the required Q-table in the time required.

Q-Learning and Deep Q-Learning

❑Why ‘deep’ Q-learning?

❑A neural network becomes helpful in approximating the Q-value function in deep Q-learning.

❑The state gets taken as the input and Q-values for all possible actions get generated as the output.

❑The following steps are involved in reinforcement learning using deep Q-learning networks (DQNs):

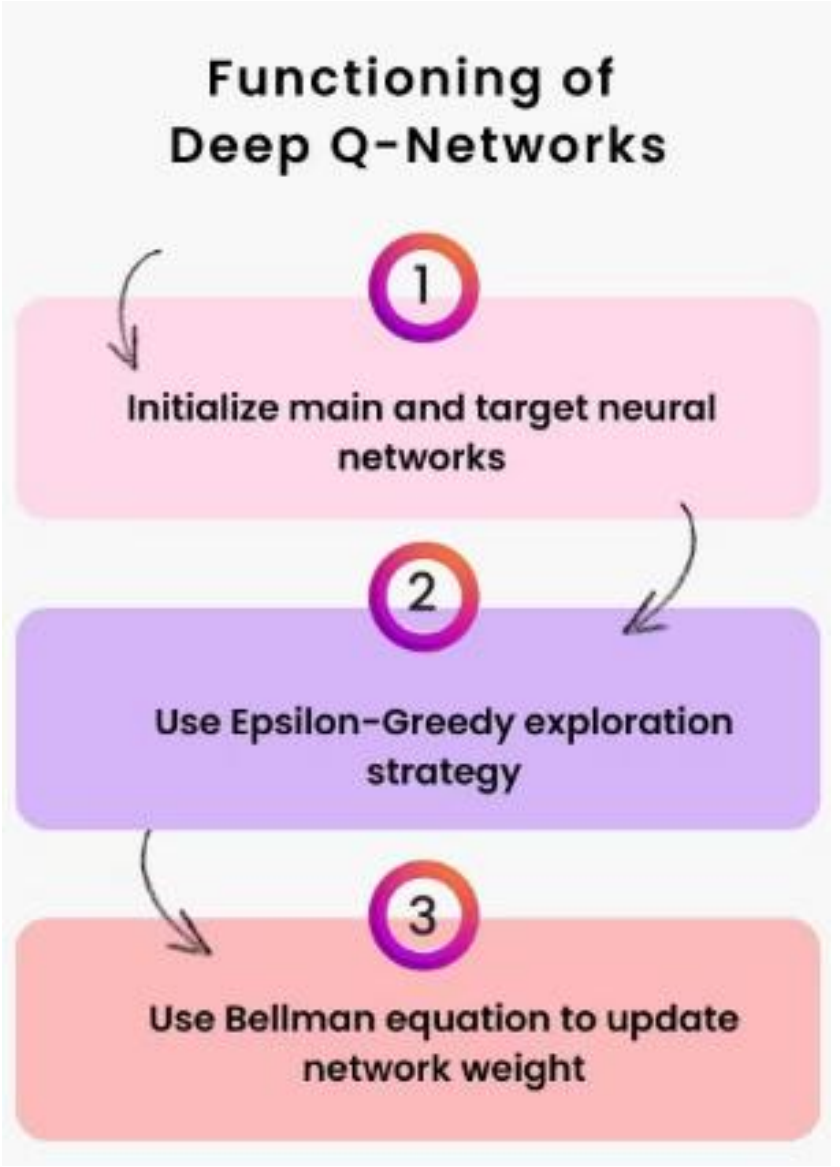
1. User memories get stored with past experiences.
2. A Q-network decides the next action based on its maximum output.
3. In the loss function, the mean square error is the difference between the target Q-value Q* and the predicted Q-value.

❑To prevent the neural network from affecting the distribution of states, actions, rewards, and next_states it encounters, deep Q-learning uses experience replay to learn in small batches.

❑Moreover, the agent doesn't need training after each step.

Q-Learning and Deep Q-Learning

❑How deep Q-networks function



Q-Learning and Deep Q-Learning

❑How deep Q-networks function:

1. Initialize main and target neural networks

❑The Q-table implementation is the prime difference between Q-learning and deep Q-learning. In the latter, neural networks replace the regular Q-table.

❑In neural networks, input states get mapped to (action, Q-value) pairs instead of state-action pairs to Q-values. A unique feature of deep Q-learning is that it uses two neural networks in its learning process.

❑Despite having the same architecture, these networks differ in their weights. During each N-step, the weights get copied between the main and target networks. Both networks help the algorithm learn more effectively and stabilize the learning process.

❑How to map states to (action, Q-value) pairs

❑An (action, Q-value) pair gets mapped between the main and target neural networks. The output nodes (representing actions) contain a floating-point value representing each action's Q-value. As the output nodes do not reflect a probability distribution, they cannot add up to 1.

❑For the example above, we have a Q-value of 8 for one action and 5 for the other.

❑How deep Q-networks function:

2. Choose an action using the Epsilon-Greedy exploration strategy

- ❑ Using the Epsilon-Greedy exploration strategy, an agent selects an action at random with probability as epsilon. It further uses its best-known action with probability equivalent to 1 epsilon.
- ❑ What are the best-known actions from a network?
- ❑ There is a mapping between input states and output actions in both the target and main models.
- ❑ These output actions correspond to the predicted Q-value for the model. When the largest Q-value gets predicted for an action, that action is the best-known at that state.

❑How deep Q-networks function:

3. Bellman equation to update network weight

❑The agents have to perform an action when they choose it and update the main and target networks using the Bellman equation.

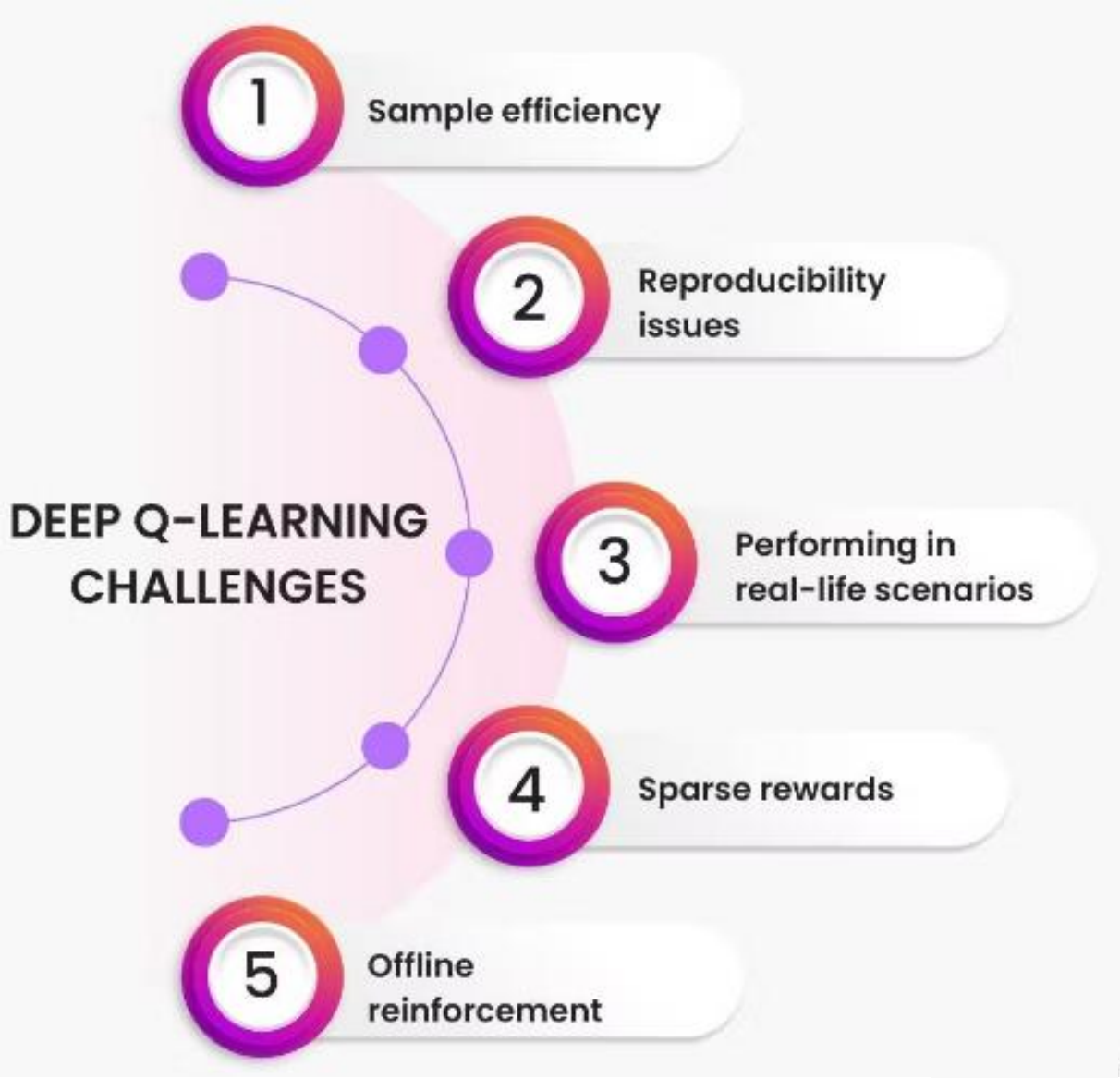
❑In order to update them, a deep Q-learning agent deploys experience replay which enables them to acquire knowledge about their environment.

❑Essentially, every step involves sampling and training which is based on a batch of 4 steps based on the past experiences.

❑After 100 such steps, the weights of the main network get transferred to the weight of the target network.

Q-Learning and Deep Q-Learning

❑Challenges in deep Q-learning



Advantage Actor Critic (A2C)

❑What is the Actor-Critic Algorithm?

❑The actor-critic algorithm is a type of reinforcement learning algorithm that combines aspects of both policy-based methods (Actor) and value-based methods (Critic).

❑This hybrid approach is designed to address the limitations of each method when used individually.

❑In the actor-critic framework, an agent (the “actor”) learns a policy to make decisions, and a value function (the “Critic”) evaluates the actions taken by the Actor.

❑Simultaneously, the critic evaluates these actions by estimating their value or quality.

❑This dual role allows the method to strike a balance between exploration and exploitation, leveraging the strengths of both policy and value functions.

Advantage Actor Critic (A2C)

❑Roles of Actor and Critic

1. **Actor:** The actor makes decisions by selecting actions based on the current policy. Its responsibility lies in exploring the action space to maximize expected cumulative rewards. By continuously refining the policy, the actor adapts to the dynamic nature of the environment.
2. **Critic:** The critic evaluates the actions taken by the actor. It estimates the value or quality of these actions by providing feedback on their performance. The critic's role is pivotal in guiding the actor towards actions that lead to higher expected returns, contributing to the overall improvement of the learning process.

Advantage Actor Critic (A2C)

❑Key Terms in Actor Critic Algorithm

There are two key terms:

- **Policy (Actor) :**
  - The policy, denoted as $\pi(a|s)$, represents the probability of taking action **a** in state **s**.
  - The actor seeks to maximize the expected return by optimizing this policy.
  - The policy is modeled by the actor network, and its parameters are denoted by θ
- **Value Function (Critic) :**
  - The value function, denoted as $V(s)$, estimates the expected cumulative reward starting from state **s**.
  - The value function is modeled by the critic network, and its parameters are denoted by **w.**

Advantage Actor Critic (A2C)

❑How Actor-Critic Algorithm works?

❑Actor Critic Algorithm Objective Function

❑The objective function for the Actor-Critic algorithm is a combination of the policy gradient (for the actor) and the value function (for the critic).

❑The overall objective function is typically expressed as the sum of two components:

Policy Gradient (Actor)

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=0}^{N} \nabla_\theta \log \pi_\theta(a_i|s_i) \cdot A(s_i, a_i)$$

Here,

- $J(\theta)$ represents the expected return under the policy parameterized by θ
- $\pi_\theta(a \mid s)$ is the policy function
- N is the number of sampled experiences.
- $A(s, a)$ is the advantage function representing the advantage of taking action **a** in state **s**.
- *i* represents the index of the sample

Advantage Actor Critic (A2C)

❑How Actor-Critic Algorithm works?

Value Function Update (Critic)

$$\nabla_w J(w) \approx \frac{1}{N} \sum_{i=1}^{N} \nabla_w (V_w(s_i) - Q_w(s_i, a_i))^2$$

Here,

- $\nabla_w J(w)$ is the gradient of the loss function with respect to the critic's parameters **w**.
- N is number of samples
- $V_w(s_i)$ is the critic's estimate of value of state **s** with parameter **w**
- $Q_w(s_i, a_i)$ is the critic's estimate of the action-value of taking action **a**
- *i* represents the index of the sample

Advantage Actor Critic (A2C)

❑How Actor-Critic Algorithm works?

❑**Update Rules:** The update rules for the actor and critic involve adjusting their respective parameters using gradient ascent (for the actor) and gradient descent (for the critic).

Actor Update

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta J(\theta_t)$$

Here,

- α: learning rate for the actor
- t is the time step within an episode

Critic Update

$$w_t = w_t - \beta \nabla_w J(w_t)$$

Here,

- w represents the parameters of the critic network
- β is the learning rate for the critic

Advantage Actor Critic (A2C)

❑How Actor-Critic Algorithm works?

Advantage Function

The advantage function, $A(s, a)$, measures the advantage of taking action **a** in state **s** over the expected value of the state under the current policy.

$$A(s, a) = Q(s, a) - V(s)$$

The advantage function, then, provides a measure of how much better or worse an action is compared to the average action.

These mathematical expressions highlight the essential computations involved in the Actor-Critic method. The actor is updated based on the policy gradient, encouraging actions with higher advantages, while the critic is updated to minimize the difference between the estimated value and the action-value.

Advantage Actor Critic (A2C)

❑A2C (Advantage Actor-Critic)

❑A2C (Advantage Actor-Critic) is a specific variant of the Actor-Critic algorithm that introduces the concept of the advantage function.

❑This function measures how much better an action is compared to the average action in a given state.

❑By incorporating this advantage information, A2C focuses the learning process on actions that have a significantly higher value than the typical action taken in that state.

❑While both leverage the actor-critic architecture, here's a key distinction between them:

1. **Learning from the Average:** The base Actor-Critic method uses the difference between the actual reward and the estimated value (critic's evaluation) to update the actor.
2. **Learning from the Advantage:** A2C leverages the advantage function, incorporating the difference between the action's value and the average value of actions in that state. This additional information refines the learning process further.

Advantage Actor Critic (A2C)

❑A2C (Advantage Actor-Critic) Algorithm Steps

The Actor-Critic algorithm combines these mathematical principles into a coherent learning framework. The algorithm involves:

1. **Initialization:**
   - Initialize the policy parameters θ ***(actor)*** and the value function parameters ϕ ***(critic)***.

2. **Interaction with the Environment:**
   - The agent interacts with the environment by taking actions according to the current policy and receiving observations and rewards in return.

3. **Advantage Computation:**
   - Compute the advantage function **A(s,a)** based on the current policy and value estimates.

4. **Policy and Value Updates:**
   - Simultaneously update the actor's parameters(θ) using the policy gradient. The policy gradient is derived from the advantage function and guides the actor to increase the probabilities of actions that lead to higher advantages.
   - Simultaneously update the critic's parameters (ϕ)using a value-based method. This often involves minimizing the temporal difference (TD) error, which is the difference between the observed rewards and the predicted values.

Asynchronous Advantage Actor Critic (A3C)

- The Asynchronous Advantage Actor Critic (A3C) algorithm is one of the newest algorithms to be developed under the field of Deep Reinforcement Learning Algorithms.
- This algorithm was developed by Google's DeepMind which is the Artificial Intelligence division of Google.
- This algorithm was first mentioned in 2016 in a research paper appropriately named Asynchronous Methods for Deep Learning.

Asynchronous Advantage Actor Critic (A3C)

❑Decoding the **different parts** of the algorithm's name:-

1. Asynchronous:

▪Unlike other popular Deep Reinforcement Learning algorithms like Deep Q-Learning which uses a single agent and a single environment.

▪This algorithm uses multiple agents with each agent having its own network parameters and a copy of the environment.

▪This agents interact with their respective environments **Asynchronously**, learning with each interaction.

▪Each agent is controlled by a global network.

▪As each agent gains more knowledge, it contributes to the total knowledge of the global network.

▪The presence of a global network allows each agent to have more diversified training data.

▪This setup mimics the real-life environment in which humans live as each human gains knowledge from the experiences of some other human thus allowing the whole "global network" to be better.

Asynchronous Advantage Actor Critic (A3C)

❑Decoding the **different parts** of the algorithm's name:-

2. Actor-Critic

Actor-Critic: Unlike some simpler techniques which are based on either Value-Iteration methods or Policy-Gradient methods, the A3C algorithm combines the best parts of both the methods ie the algorithm predicts both the value function V(s) as well as the optimal policy function $\pi(s)$. The learning agent uses the value of the Value function (Critic) to update the optimal policy function (Actor). Note that here the policy function means the **probabilistic distribution of the action space**. To be exact, the learning agent determines the conditional probability P(a|s ; θ) ie the parameterized probability that the agent chooses the action a when in state s.

Asynchronous Advantage Actor Critic (A3C)

❑Decoding the **different parts** of the algorithm's name:-

3. Advantage:

▪Typically in the implementation of **Policy Gradient**, the value of Discounted Returns() to tell the agent which of it's actions were rewarding and which ones were penalized.

▪By using the value of Advantage instead, the agent also learns how much better the rewards were than it's expectation.

▪This gives a new-found insight to the agent into the environment and thus the learning process is better.

▪The advantage metric is given by the following expression:-

$$\textbf{Advantage: } A = Q(s, a) - V(s)$$

Note for Students

- This power point presentation is for lecture, therefore it is suggested that also utilize the text books and lecture notes.