Assignment

**Query 1: Can you write Deep Q Learning Algorithm using pure Python without any libraries?**

**Answer 1:**

Deep Q-Learning is a reinforcement learning technique that combines Q-Learning, an algorithm for learning optimal actions in an environment, with deep neural networks.

Deep Q-Learning has proven effective in training agents for various tasks, including video games and robotic control.

import numpy as np

import random

class DeepQLearning: #Define the DeepQLearning

def \_\_init\_\_(self, state\_space, action\_space, epsilon, gamma, learning\_rate):

self.state\_space = state\_space

self.action\_space = action\_space

self.epsilon = epsilon

self.gamma = gamma

self.learning\_rate = learning\_rate

self.q\_table = np.zeros((len(state\_space), len(action\_space)))

def get\_action(self, state):

if random.random() < self.epsilon:

return random.choice(action\_space)

return np.argmax(self.q\_table[state])

def update\_q\_table(self, state, action, reward, next\_state):

self.q\_table[state][action] = (1 - self.learning\_rate) \* self.q\_table[state][action] + \

self.learning\_rate \* (reward + self.gamma \* np.max(self.q\_table[next\_state]))

def train(self, episodes): # Define train function

for episode in range(episodes):

state = np.random.choice(state\_space)

action = self.get\_action(state)

reward, next\_state = self.env.step(action)

self.update\_q\_table(state, action, reward, next\_state)

def test(self, episodes): #Define test function

for episode in range(episodes):

state = np.random.choice(state\_space)

action = self.get\_action(state)

reward, next\_state = self.env.step(action)

if reward > 0: #Define Rewards

print("The agent won!")

else:

print("The agent lost.")

if \_\_name\_\_ == "\_\_main\_\_":

env = gym.make("CartPole-v0")

agent = DeepQLearning(env.observation\_space, env.action\_space, 0.1, 0.9, 0.01)

agent.train(1000)

agent.test(100)