**Information File: Training an Agent in a 5x5 2D Custom Environment**

**Goal:**

The goal of this simulation is to train an agent in a custom-built 5x5 grid environment to reach a target goal (state 24) using Q-learning. The environment has a simple grid layout, and the agent can take four possible actions: move up, down, left, or right. The Q-learning algorithm is used to help the agent learn the optimal path to reach the goal, while minimizing exploration as the agent becomes more knowledgeable about the environment.

**Concepts:**

**1. Custom Environment (Ground class):**

* A 5x5 grid where the agent starts at the top-left corner (state 0) and aims to reach the bottom-right corner (state 24).
* The agent has four possible actions: move up (0), down (1), left (2), or right (3).
* Rewards:
  + +1 for reaching the goal state (state 24).
  + -1 for other states (penalizing steps not leading to the goal).
* The agent has a limited number of steps (200 steps) to reach the goal before the episode terminates.

**2. State Space:**

* A discrete representation of the environment, with states represented by numbers from 0 to 24 (5x5 grid).
* The state space consists of all the possible positions the agent can occupy in the grid.

**3. Action Space:**

* The possible moves an agent can make from a given state.
* In this environment, there are four actions: move up, down, left, or right.

**4. Q-Learning Algorithm:**

* A model-free reinforcement learning algorithm used to train the agent.
* The Q-learning formula used to update the Q-values is: Q(s,a)=Q(s,a)+α(r+γmax⁡(Q(s′,a′))−Q(s,a))Q(s, a) = Q(s, a) + \alpha \left( r + \gamma \max(Q(s', a')) - Q(s, a) \right)Q(s,a)=Q(s,a)+α(r+γmax(Q(s′,a′))−Q(s,a)) Where:
  + Q(s,a)Q(s, a)Q(s,a) is the Q-value of the current state-action pair.
  + α\alphaα is the learning rate (0.9 in this case).
  + rrr is the reward for taking action aaa from state sss.
  + γ\gammaγ is the discount factor for future rewards (0.9).
  + max⁡(Q(s′,a′))\max(Q(s', a'))max(Q(s′,a′)) is the maximum Q-value of the next state.

**5. Exploration vs Exploitation (Epsilon-Greedy Policy):**

* The agent selects actions based on an epsilon-greedy policy:
  + With probability ϵ\epsilonϵ, the agent explores by selecting a random action.
  + With probability 1−ϵ1 - \epsilon1−ϵ, the agent exploits by selecting the action with the highest Q-value.
* Epsilon decays over time, reducing exploration and increasing exploitation as the agent learns the environment.

**6. Hyperparameters:**

* **Learning Rate (α\alphaα):** Controls how much new information overrides the old knowledge. In this case, α=0.9\alpha = 0.9α=0.9.
* **Discount Factor (γ\gammaγ):** Represents the importance of future rewards. In this case, γ=0.9\gamma = 0.9γ=0.9.
* **Exploration Rate (ϵ\epsilonϵ):** Determines the probability of exploring versus exploiting. It starts at 1.0 and decays over time to a minimum of 0.1, controlled by the decay factor (0.99).

**7. Rewards Plot:**

* A graph plotting the cumulative rewards over episodes.
* Shows how the agent's performance improves over time, with initial exploration and eventual exploitation of the optimal path to the goal.