SIS420 - I.A.

Alumno: Delgadillo Llanos Juan Sebastian

Implementación de AxR con el entorno Gymnasium .

Repositorio: GITHUB

Informe del código: Algoritmo de Aprendizaje Q-Learning para el entorno Taxi-v3 de Gymnasium

Descripción General

PRIMERO ("AxR_Taxi.py"):

El código implementa un algoritmo de aprendizaje Q-learning para entrenar un agente en el entorno Taxi-v3 de Gymnasium. El objetivo del agente es aprender la mejor política para maximizar la recompensa acumulada al completar tareas de recoger y dejar pasajeros en ubicaciones específicas dentro de una cuadrícula, una vez entrenado se mostrará una imagen de las recompensas obtenidas a lo largo del entrenamiento.

SEGUNDO ("AxR_Taxi_pkl.py"):

El código implementa un algoritmo de aprendizaje Q-learning para entrenar un agente en el entorno Taxi-v3 de Gymnasium. El objetivo del agente es aprender la mejor política para maximizar la recompensa acumulada al completar tareas de recoger y dejar pasajeros en ubicaciones específicas dentro de una cuadrícula, una vez entrenado se guardará la tabla Q en un archivo "taxi.pkl", el cual dependiendo del modo de ejecución se explorará o se explotará, también se guardará una imagen "taxi.png" de las recompensas obtenidas a lo largo del entrenamiento.

Algoritmo Utilizado: Q-Learning

El algoritmo Q-learning es un método de aprendizaje por refuerzo que busca aprender la función Q, la cual estima la calidad de una acción en un estado determinado. Este método se basa en la ecuación de Bellman, que actualiza los valores de la función Q iterativamente:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left(r + \gamma \max_{a'} Q(s',a') - Q(s,a) \right)$$

donde:

- $Q(s,a)$ es el valor actual de la función Q para el estado s y la acción a.
- α es la tasa de aprendizaje.
- r es la recompensa obtenida después de tomar la acción a.
- γ es el factor de descuento que valora las recompensas futuras.
- s' es el nuevo estado después de tomar la acción a.
- $\max_{a'} Q(s',a')$ es el valor de la mejor acción posible en el nuevo estado s'.

Estructura del Código

Importación de Librerías:

```
import gymnasium as gym
import numpy as np
import matplotlib.pyplot as plt
```

Función de Entrenamiento e inicialización del Entorno:

```
def train(episodes): env

= gym.make("Taxi-v3")
```

Inicialización de la Tabla Q y definición de Parámetros:

```
q_table = np.zeros((env.observation_space.n, env.action_space.n))

learning_rate = 0.3
discount_factor = 0.9
epsilon = 1.0
epsilon_decay_rate = 0.0003
rng = np.random.default_rng()
```

Bucle de Entrenamiento y reinicio del entorno:

```
state = env.reset()[0]
```

```
for i in range(episodes):


if (i + 1) % 100 == 0:
env.close()
    env = gym.make("Taxi-v3",
render_mode="human") else:     env.close()
    env = gym.make("Taxi-v3")
```

Bucle de Pasos dentro de un Episodio:

```
while not terminated and not truncated:
if rng.random() < epsilon:         action =
env.action_space.sample()     else:
action = np.argmax(q_table[state, :])
    new_state, reward, terminated, truncated, _ = env.step(action)
    q_table[state, action] = q_table[state, action] + learning_rate * (reward +
discount_factor * np.max(q_table[new_state, :]) - q_table[state, action])
    state = new_state
```

Actualización de Epsilon:

```
epsilon = max(epsilon - epsilon_decay_rate, 0.01)
rewards_por_episode[i] = reward
```

Visualización de Resultados:

```
suma_rewards = np.zeros(episodes) for t in range(episodes):
suma_rewards[t] = np.sum(rewards_por_episode[max(0, t - 100) :(t + 1)])
plt.plot(suma_rewards)
plt.xlabel('Episodios')
plt.ylabel('Suma de recompensas acumuladas')
plt.title('Evolución de las recompensas acumuladas durante el entrenamiento')
plt.show()
```

Conclusiones

El código implementa el algoritmo Q-learning para entrenar un agente en el entorno Taxi-v3. Utiliza una tabla Q para almacenar los valores de calidad de las acciones en cada estado, que se actualizan iterativamente mediante la ecuación de Bellman. La estrategia epsilon-greedy permite un balance entre exploración y explotación, ajustándose con el tiempo para mejorar la política del agente. Los resultados se visualizan al final del entrenamiento para evaluar el rendimiento del agente.