گزارش پروژه سوم

پروژه هزارتو

درس: مبانی و کاربردهای هوش مصنوعی

استاد راهنما: دکتر حسین کارشناس نجف آبادی

اعضای گروه:

علی‌اکبر احراری- ۴۰۰۳۶۱۳۰۰۱

مهرآذین مرزوق- ۴۰۰۳۶۱۳۰۵۵

پاییز ۱۴۰۲

فهرست

گزارش کار الگوریتم

ساخت محیط

ابتدا با استفاده از کتابخانه Gym به ساخت محیط کلی پروژه می‌پردازیم. در کد بعدی برنامه به وسیله تابع env.reset()، محیط برنامه را به صورت مجدد تنظیم کرده و مقدار بازگشته که حالت اولیه می‌باشد را درون observation ذخیره می‌کنیم.

```
env = gym.make("maze-random-10x10-plus-v0")
observation = env.reset()
```

ساخت q-table و مقداردهی اولیه پارامترها

جدول Q که متشکل از یک جدول 10 در ۱۰ می‌باشد را مقداردهی اولیه می‌کنیم. همچنین تعداد عملیات ممکن(Actions) را داخل متغیر num_actions ذخیره می‌کنیم.

در ادامه پارامترهای مورد نیاز در برنامه را مقداردهی اولیه می‌کنیم.

```
        # Q-learning parameters
alpha = 0.1  # Learning rate
gamma = 0.9  # Discount factor
epsilon = 0.99  # Epsilon-greedy parameter
min_epsilon = 0.1  # Minimum epsilon value
epsilon_decay = 0.99  # Epsilon decay rate

NUM_EPISODES = 1000  # Maximum episodes
total_finds = 0  # Total number of times the agent finishes the maze
episodes = list()   # Episodes agent finished the maze
```

عملیات اصلی

با ورود به حلقه اصلی برنامه، ابتدا محیط را راه‌اندازی مجدد می‌کنیم. مقادیری را نیز برای دنبال کردن مجموع پاداش، قدم‌ها و همچنین بررسی به پایان رسیدن قسمت‌ها درنظر می‌گیریم. در ادامه وارد حلقه دوم برنامه شده و با env.render() محیط برنامه را به صورت بصری مجسم می‌کنیم. با استفاده از Q-values به‌دست آمده از جدول Qها، به صورت حریصانه به یافتن بهترین حرکت بعدی می‌پردازیم.

```
        for episode in range(NUM_EPISODES):

                state = env.reset()
                state = int(state[0] * 10 + state[1])

                total_reward = 0
                steps = 0
                done = False

                while not done and steps <= 100:
                        env.render()

                action = np.argmax(Q[state])  # Greedy action based on Q-
values
```

سپس با استفاده از تابع step و قراردادن action به‌دست آمده به‌عنوان ورودی این تابع، حرکت مورد نظر را عملی می‌کنیم. سپس حالت بعدی برنامه را پیدا می‌کنیم. حال با استفاده از آن‌ها، جدول Q را با استفاده از فرمول Q-learningبه‌روزرسانی می‌کنیم. در ادامه با استفاده از یک شرط، بررسی می‌کنیم آیا عامل به‌صورت موفقیت‌آمیز به هدف رسیده یا خیر. در صورت رسیدن موفقیت‌آمیز، آن قسمت را به لیست قسمت‌های موفق می‌افزاییم. پس از بررسی شرط، یک قدم به جلو می‌رویم.

```
# Perform the chosen action
next_state, reward, done, _ = env.step(action)
next_state = int(next_state[0] * 10 + next_state[1])

# Update Q-value for the state-action pair
Q[state][action] += alpha * (reward + gamma * np.max(Q[next_state]) -
Q[state][action])

total_reward += reward
state = next_state

if done:
    total_finds += 1
    episodes.append(episode)
    break

steps += 1
```

Epsilon پارامتر اکتشاف- بهره‌برداری را با تجزیه آن پس از هر قسمت بر اساس میزان decay-rate (Epsilon_decay) به‌روزرسانی می‌کند و با پیشرفت آموزش، تغییر به سمت exploitationرا تضمین می‌کند.

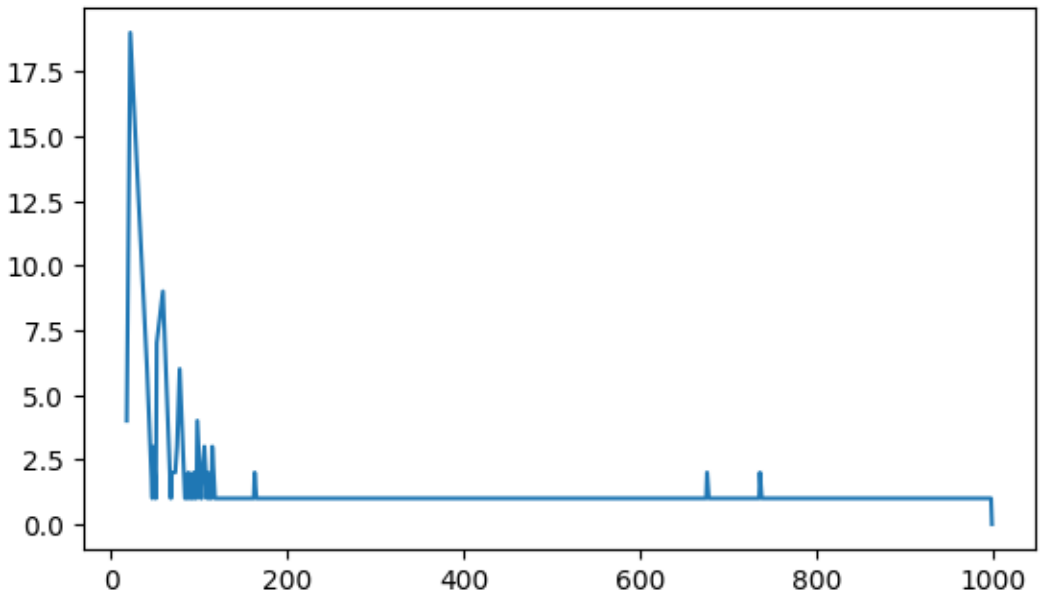
```
# Epsilon decay after each episode
epsilon = max(min_epsilon, epsilon * epsilon_decay)
```

رسم نمودار

در بخش نهایی پروژه، به رسم یک نمودار می‌پردازیم که در آن به نمایش تعداد قدم‌های منجربه برد در هر قسمت می‌پردازیم. Yها نشان‌دهنده تعداد قدم‌های برداشته شده و Xها نشان‌دهنده هر قسمت می‌باشند. در پایان برنامه را می‌بندیم.

```
y = list()
for i in range(len(episodes) - 1):
    y.append(episodes[i + 1] - episodes[i])
y.append(0)
x = episodes
plot.plot(x, y)
plot.show()
# Close the environment
env.close()
```

نمونه‌ای از خروجی



Winning rate: 0.916

کتابخانه‌های استفاده شده

Numpy: انجام محاسبات عددی و کار با آرایه‌ها

Matplotlib: رسم نمودار

Gym و Gym_maze: ساخت محیط مناسب

منابع

An introduction to Q-Learning: Reinforcement Learning (floydhub.com)

An Introduction to Q-Learning: A Tutorial For Beginners | DataCamp

Bing Chat with GPT-4

Artificial Intelligence: A Modern Approach, Textbook by Peter Norvig and Stuart J. Russell