Áp dụng Reinforcement Learning vào Car Racing Game

Thành viên nhóm

Vũ Quốc Khánh – 17021275
Email: 17021275@vnu.edu.vn
Nguyễn Việt Hoàng – 17021257
Email: 17021257@vnu.edu.vn

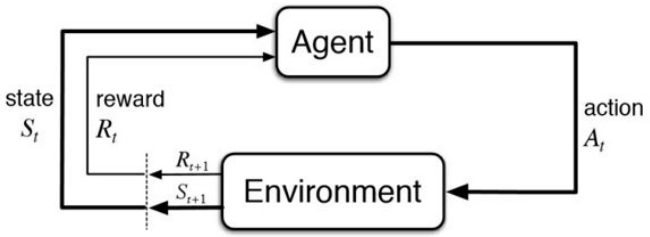
1. Nhiệm vụ

- Reinforcement Learning hay học củng cố/tăng cường, là lĩnh vực liên quan đến việc dạy cho máy (agent) thực hiện tốt một nhiệm vụ (task) bằng cách tương tác với môi trường (environment) thông qua hành động (action) và nhận được phần thưởng (reward). Cách học này rất giống với cách con người học từ môi trường bằng cách thử sai. Lấy ví dụ 1 đứa vào mùa đông đến gần lửa thì thấy ấm, đứa trẻ sẽ có xu hướng đến gần lửa nhiều hơn (vì nhận được phần thưởng là ấm áp), nhưng chạm vào lửa nóng, đứa trẻ sẽ có xu hướng tránh chạm vào lửa (vì bị là bỏng tay).

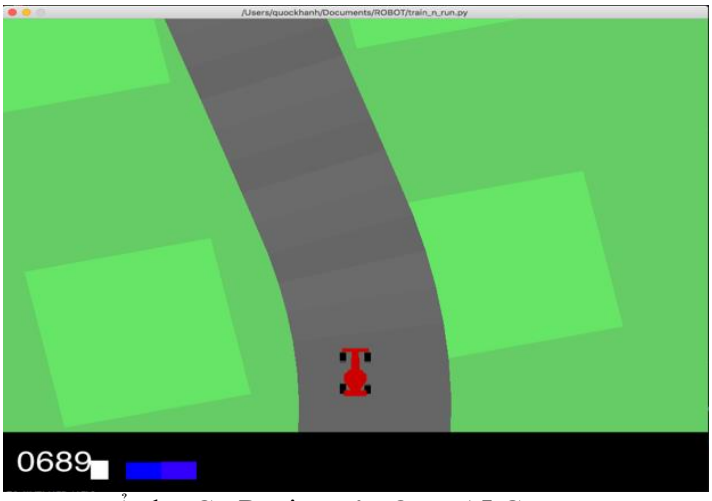
Không như unsupervised learning là tìm ra điểm giống và khác nhau giữa các Sample, nhiệm vụ của Reinforcement learning là tìm ra Actions phù hợp để maximize Reward của Agent. Dưới đây là mô tả tương tác giữa Agent – Enviroment.

- Dưới đây là định nghĩa của các thuật ngữ hay xuất hiện trong RL:

- *Environment* (môi trường): là không gian mà máy tương tác.
- *Agent* (máy): máy quan sát môi trường và sinh ra hành động tương ứng.
- *Policy* (chiến thuật): máy sẽ theo chiến thuật như thế nào để đạt được mục đích.
- *Reward* (phần thưởng): phần thưởng tương ứng từ môi trường mà máy nhận được khi thực hiện một hành động.
- *State* (trạng thái): trạng thái của môi trường mà máy nhận được.
- *Episode* (tập): một chuỗi các trạng thái và hành động cho đến trạng thái kết thúc s1,a1,s2,a2,...sT,aT
- *Accumulative Reward* (phần thưởng tích lũy): tổng phần thưởng tích lũy từ 1 state đến state cuối cùng. Như vậy, tại state s, agent tương tác với environment với hành động a, dẫn đến state mới st+1 và nhận được reward tương ứng rt+1. Vòng lặp như thế cho đến trạng thái cuối cùng sT.



Ảnh : Basic Reinforcement Learning framework



Ảnh : CarRacing của OpenAI Gym

- Nhiệm vụ của nhóm là tìm hiểu và áp dụng mô hình Deep Q-learning vào game CarRacing_v0 của thư viện OpenAI Gym.

2. Khái niệm

2.1 Deep Q Network (DQN)

- Deep Q Network (DQN) là một trong những giải thuật Học tăng cường nổi tiếng gần đây, ý tưởng cốt lõi của giải thuật này là Q-learning (Watkins and Dayan, 1992). Giải thuật Q-Learning kinh điển dựa trên Hàm xấp xỉ Q(st,at)=𝔼[Rt∥st,at]Q(st,at)=E[Rt∥st,at], Hàm này sẽ dự đoán Phần thưởng tương lai cao nhất khi ta thực hiện một hành động a tại trạng thái s. Trong giải thuật gốc, Hàm xấp xỉ Q này được thiết kế là một Bảng gọi là Q-Table, trong đó Hàng là các trạng thái có thể có, cột là các Hành động trong game. Trong quá trình học, bảng này

sẽ lưu lại các giá trị Phần thưởng tương lai cao nhất cho mỗi hành động tại 1 trạng thái nhất định.
- Tuy nhiên với thiết kế này khi áp dụng cho game gặp phải vấn đề về lưu trữ Trạng thái trong Q-Table. Tập trạng thái quá lớn để lưu trữ hết trên RAM. Do đó Deep Q-Network ra đời, DQN sẽ thay thế Q-table bằng mạng Nơ ron đủ tốt để có thể dự đoán Phần thưởng cho mỗi hành động tại một trạng thái nhất định. Khi mạng Nơ ron này đủ tốt nó cho phép ta tính toán được Phần thưởng tốt nhất khi thực hiện một hành động tại một Trạng thái bất kỳ. Hàm xấp xỉ lúc này được gọi là Phương trình Bellman như sau:

$$Q(st,at)=rt+\gamma maxat+1Q(st+1,at+1)$$

- Trong đó:
  S: Trạng thái
  a: Hành động thực hiện
  r: Phần thưởng hiện tại
  gamma: Toán tử giảm

- Để mạng Nơ ron hội tụ được ta cần hàm lỗi (Loss) để điều chỉnh trọng số mạng:

$$L=(r+\gamma maxat+1Q(st+1,at+1)-Q(st,at))2$$

Qua thời gian học, mạng sẽ dần dần trở nên chính xác

2.2 Experience replay

- Phần trên ta đã định nghĩa một mạng Nơ ron lấy input là state hiện tại và output các Q-value. Thế nhưng nếu mạng Nơ ron cứ liên tục bị đẩy vào từng state một sẽ rất dễ bị overfitting vì các states liên tục thường giống nhau hoặc có tính tuyến tính (ví dụ: liên tục đi thẳng/sang trái/phải). Kỹ thuật Experience Replay được sử dụng để loại bỏ vấn đề này.
- Thay vì mỗi state mạng update một lần, ta lưu lại các states vào bộ nhớ (memory). Sau đó thực hiện sampling thành các batch nhỏ đưa vào NN học. Việc này giúp đa dạng hóa input và tránh bị overfitting.

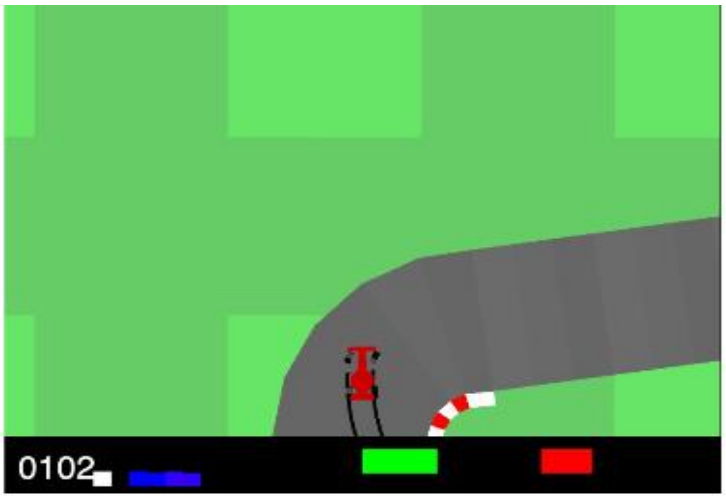
2.3 Nhìn lại toàn bộ mô hình

- Tóm lại, Deep Q-Learning thực hiện các bước sau:

B1: Enviroment đưa vào mạng một state s; đầu ra là các Q-value của các actions tương ứng.
B2: Agent chọn action bằng một Policy và thực hiện action đó.
B3: Environment trả lại state s' và reward r là kết quả của action a và lưu experience tuple [s, a, r, s'] vào memory
B4: Thực hiện sample các experience thành một vài batches và tiến hành train NN
B5: Lặp lại đến khi kết thúc M episodes

3. Enviroment

- Nhóm sử dụng game CarRacing_v0 từ thư viện OpenAI Gym làm Enviroment.
- Agent cho sẵn là xe đua màu đỏ trong game. Mỗi state là một screenshot được chụp mỗi 4 frames 96x96 pixels. Có 4 actions: Đi tiến, phanh, rẽ trái và rẽ phải. Reward sẽ -0.1 mỗi frame và +1000/N mỗi track tile được visit, trong đó N là tổng số tiles trong đường đua. Nếu xe đua đi lệch khỏi đường đua và ra ngoài cỏ thì reward sẽ bị -100. Ví dụ, nếu đã hoàn thành 732 frames, reward sẽ là 1000 – 0.1*732 = 926.8 điểm. Mỗi episodes sẽ được hoàn thành khi tất cả các tiles được visit hoặc xe bị lệch xa khỏi đường đua.



Ảnh: Reward và các chỉ số

- Khi enviroment được bắt đầu, các chỉ số được show ở dưới cùng của cửa sổ. Từ trái qua phải: Điểm số, tốc độ (cột trắng), phanh (xanh), vị trí tay lái (xanh lá cây) và con quay hồi chuyển (đỏ). Trong khi game được chạy, action của xe sẽ được in ra terminal mỗi 200 steps hoặc khi xe

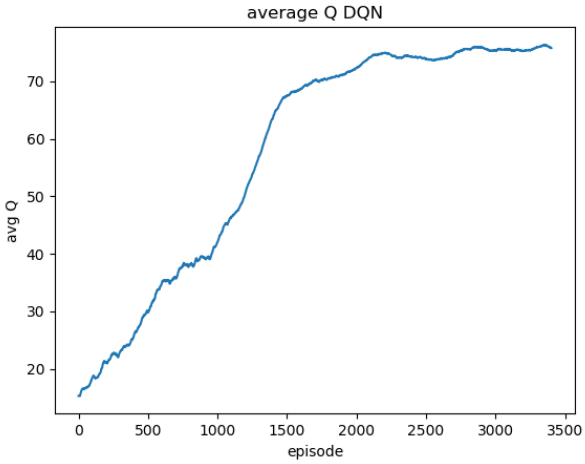
đi lệch khỏi đường đua hoặc một episodes được hoàn thành. Một Actions map bao gồm 3 phần. Giá trị đầu tương ứng với trái hoặc phải. Nếu trái thì = -1, phải = +1. Giá trị thứ 2 cho biết đi tiến hay lùi. Nếu tiến thì là +1. Giá trị thứ 3 là phanh. Nếu phanh thì +0.8.

```
action['+0.00', '+0.00', '+0.80']
step 1200 total_reward -68.44

action['+1.00', '+0.00', '+0.00']
step 1400 total_reward -88.44

action['+0.00', '+1.00', '+0.00']
step 1600 total_reward -108.44
```

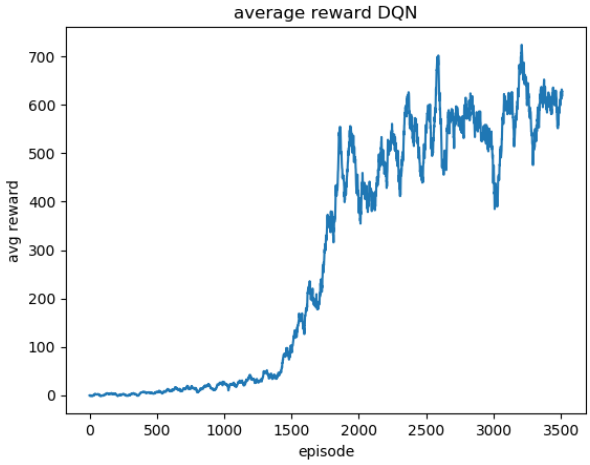
Ảnh : Actions map

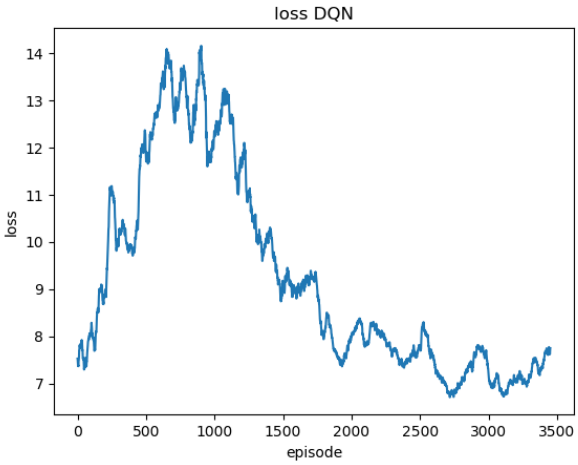
4. Kết quả

- Trong quá trình thực nghiệm, nhóm đã train model ở 2 máy khác nhau, mỗi máy được train trong vòng 10 tiếng và ghi lại giá trị trung bình của Q, giá trị trung bình của reward và giá trị loss mỗi episode. Nhóm đã đặt tổng số lần train là 5000 episode. Đây là con số đủ lớn để có thể hội tụ được.
- Đồ thị dưới đây là kết quả training sau 5000 episodes. Model tiến tới trạng thái ổn định sau



1500 episodes. Giá trị Q trung bình được giữ ở 75 và reward trung bình vào khoảng 550. Giá trị loss tiến tới cao nhất trong khoảng 1000 episodes đầu và giảm sau đó.





- Training After 600 Episodes

Model đã biết cách đi thẳng để nhận reward. Tuy nhiên chưa biết cách rẽ khi gặp các ngã rẽ.

- Training After 1800 Episodes

Model đã học được cách rẽ. Nhưng khi gặp những ngã rẽ gấp thì model sẽ bị lệch khỏi đường.

- Training After 2000 Episodes

Model đã chạy tốt và hoàn thành hết 1 vòng đua. Tuy nhiên vẫn còn vài episodes model chạy lệch khỏi đường đua.

- Training After 3200 Episodes

Model đã chạy tốt và hoàn thành hết 1 vòng đua. Tuy nhiên vẫn còn vài episodes model chạy lệch khỏi đường đua.

- Training After 4000 Episodes

Model đã chạy tốt và reward trung bình là 850.

5. Setup

5.1. Thư viện cần thiết

- Python 3.5
- Open AI Gym (car_racing enviroment)
- Numpy
- Itertools
- scikit-image
- Tensorflow version 1.15 (hiện tại đã có bản mới nhất là version 2.0 nhưng do bị remove một số hàm cần thiết nên nhóm dùng version 1.15)

5.2. Hướng dẫn train agent và chạy thử.

- Sử dụng bàn phím
  Chạy file run_game_with_keyboard.py
- Train agent
  Chạy file train_n_run.py

6. Vai trò chi tiết

Công việc	Người làm	Đóng góp
Tạo Enviroment. Thực hiện việc chạy và training model	Vũ Quốc Khánh	60%
Thông kê dữ liệu kết quả thu được. Viết project report.	Nguyễn Việt Hoàng	40%

7. Refferences

1) Youtube RL Course by David Silver - Lecture 6: Value Function Approximation
2) Sách Hands-On Reinforcement Learning With Python by Sudharsan Ravichandiran
3) Youtube Creating a Custom OpenAI Gym Environment for your own game!
4) https://www.cs.toronto.edu/~vmnih/docs/dqn.pdf - Playing Atari with Deep Reinforcement Learning
5) https://web.stanford.edu/class/psych209/Readings/MnihEtAlHassibis15NatureControlDeepRL.pdf - Human-level control through deep reinforcement learning
6) https://kipalog.com/posts/Lam-quen-voi-OpenAI-gym-thong-qua-game-Taxi-v2 - Làm quen với OpenAI Gym thông qua game Taxi_v2
7) https://forum.machinelearningcoban.com/t/tutorial-implement-cac-thuat-toan-reinforcement-learning-ddpg-bai-1/3276 – Tutorial Implement các thuật toán Reinforcement Learning