Report for AI3007: Group 13

1st Hồ Cảnh Quyền
22022629

2nd Hà Kim Dương
22022621

3rd Phạm Anh Quân
22022625

Giảng viên hướng dẫn:Tạ Việt Cường

***Tóm tắt nội dung*—MAGent2 là một nền tảng mô phỏng đa tác nhân (multi-agent) được thiết kế để hỗ trợ nghiên cứu và phát triển các thuật toán học tăng cường (Reinforcement Learning - RL) trong các môi trường có nhiều tác nhân. Đây là một công cụ mạnh mẽ cho các nhà nghiên cứu, đặc biệt khi muốn triển khai các bài toán có sự tương tác phức tạp giữa các tác nhân.**

1. Giới thiệu

1.1. Động lực

Trong thời đại công nghệ 4.0, trí tuệ nhân tạo (AI) và học máy (Machine Learning) đã trở thành một phần không thể thiếu trong việc phát triển các hệ thống tự động hóa và tối ưu hóa trong nhiều lĩnh vực, từ công nghiệp, y tế, cho đến tài chính và giao thông.

Học tăng cường là một lĩnh vực của học máy, trong đó một tác nhân học cách tương tác với môi trường thông qua các hành động, nhận các phản hồi dưới dạng phần thưởng (reward) hoặc hình phạt (penalty), và từ đó học cách tối ưu hóa chiến lược của mình. RL đã được áp dụng thành công trong nhiều ứng dụng, bao gồm chơi game (chẳng hạn như AlphaGo của Google), tự lái xe, robot....

Chính vì vậy, học tăng cường (Reinforcement Learning) luôn là một chủ đề hấp dẫn, có ý nghĩa lớn đối với sự phát triển của công nghệ, đặc biệt trong việc tối ưu hóa các hệ thống tự động và tạo ra các tác nhân thông minh có khả năng học hỏi và ra quyết định trong môi trường phức tạp.

1.2. Mục tiêu của dự án

Mục tiêu chính là phát triển và huấn luyện một tác nhân học tăng cường sử dụng nền tảng MAgent2 để giải quyết một nhiệm vụ trong môi trường "battle"(chiến đấu). Tác nhân được đào tạo sẽ đối đầu với ba loại đối thủ khác nhau:

- **R**andom Agents: Các tác nhân thực hiện hành động ngẫu nhiên trong môi trường.
- **A** Pretrained Agent: Một tác nhân được đào tạo trước được cung cấp trong kho lưu trữ.
- **A** Final Agent: Tác nhân này được huấn luyện trong quá trình tự chơi trong khoảng thời gian dài, và sẽ là đối thủ mạnh nhất.

2. Tổng quan MAgent2

- **C**hế độ quan sát: Tác nhân quan sát một khu vực xung quanh mình, được biết dù trống, có vật cản, hoặc có tác nhân khác. Nếu một nhân vật nằm trong ô, giá trị tại ô đó biểu thị là (HP của tác nhân / HP tối đa).
- **V**ector đặc trưng: Vectơ đặc trưng chứa thông tin về chính tác nhân. Ở chế độ bình thường, nó chứa . Ở chế độ bản đồ nhỏ, nó cũng chứa vị trí tác nhân trên bản đồ, được chuẩn hóa thành 0-1.
- **Q**uan sát: Quan sát là chế độ xem quan sát 3D được nối với vectơ đặc điểm 1D bằng cách lặp lại giá trị của đặc điểm trên toàn bộ kênh hình ảnh.
- **C**hế độ bản đồ nhỏ: Cung cấp thêm thông tin toàn cục như bản đồ mật khẩu của hai đội và vị trí tuyệt đối của nhân viên trên bản đồ. Thông tin này được thêm vào quan sát 3D và vector cụ thể.Chế độ bật/tắt tùy chỉnh này thông qua tham số minimap mode
- **D**i chuyển và tấn công: Một tác nhân có thể di chuyển hoặc tấn công theo từng bước, do đó không gian hành động là sự kết nối của tất cả các động thái có thể có và tất cả các đòn tấn công có thể có.
- **S**tate:Khi gọi env.state(), bạn có thể truy xuất toàn bộ không gian quan sát 3D của môi trường, với kích thước bằng bản đồ. Không gian này bao gồm kênh đầu tiên hiển thị các bức tường, trong đó mỗi phần tử biểu thị tọa độ trống hay có chướng ngại vật. Các kênh tiếp theo cung cấp thông tin về vị trí của các tác nhân, với tỉ lệ HP hiện tại so với HP tối đa. Nếu có tính năng bổ sung, một vectơ tính năng sẽ được thêm vào và phân loại theo từng tác nhân.

3. Thực nghiệm

3.1. Cài đặt để chạy trên Colab

Các bước thực hiện: - Truy cập vào Google Colab và mở một notebook mới. - Kết nối với GPU T4 mạnh mẽ: - Clone repository của chúng tôi từ link GitHub về qua lệnh:

```
!git clone https://github.com/quyencanh203/RL-final.git
```

- Cài đặt các thư viện cần thiết (đã được tổng hợp trong file `requirements.txt`):

```
!pip install -r requirements.txt
```

- Thực hiện chạy mã với file $\texttt{train}_{a}gent_{r}edpt.py$:

```
!python train_agent_redpt.py
```

Bảng I
CÁC THAM SỐ HUẤN LUYỆN ĐƯỢC SỬ DỤNG TRONG MÔ HÌNH.

Tham số	Loại	Giá trị mặc định
Learning Rate	float	1×10^{-4}
Gamma (Discount Factor)	float	0.99
Epsilon Start	float	1.0
Epsilon End	float	0.1
Epsilon Decay Steps	int	10000
Batch Size	int	64
Target Update Frequency	int	100
Max Episodes	int	150
Replay Buffer Size	int	10000

3.2. Các tham số dùng để huấn luyện

Mô hình chạy trong thời gian là 1h10p với 150 episodes. Thực hiện chạy với GPU T4 của google colab. Sau khi chạy ta có kết quả được trình bày ở phần dưới.

3.3. Kết quả

Bảng II
WIN RATES FOR DIFFERENT MODELS.

Model	Win Rate (Red)	Win Rate (Blue)
Random	0.0	1.0
Trained Policy	1.0	0.0
Final Trained Policy	0.967	0.0

Bảng III
AVERAGE REWARDS FOR DIFFERENT MODELS.

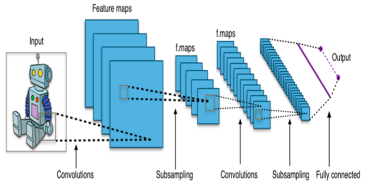
Model	Avg Rewards (Red)	Avg Rewards (Blue)
Random	-1.5161	3.5923
Trained Policy	4.7925	2.3538
Final Trained Policy	4.1843	2.4610

4. Phương pháp

Sử dụng phương pháp deep-Qlearning. Đây là một kiến trúc mạng nơ-ron có sự kết hợp giữa các lớp Convolutional Neural Networks (CNN) và các lớp Fully Connected (FC) để giải quyết bài toán học tăng cường, với mục tiêu học các giá trị Q để đưa ra các quyết định hành động trong môi trường.

4.1. Kiến trúc CNN

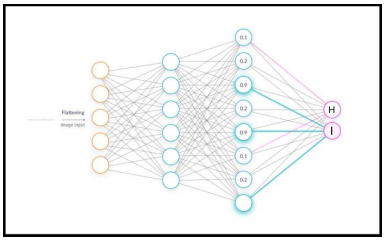
Mạng bắt đầu bằng một chuỗi các lớp CNN (Convolutional Layers), mục đích của các lớp này là trích xuất các đặc trưng không gian từ dữ liệu đầu vào. Cụ thể, các lớp này có nhiệm vụ tiếp nhận đầu vào là hình ảnh (hoặc tensor có cấu trúc không gian), thực hiện các phép toán chập với các bộ lọc (kernel) để phát hiện các đặc trưng từ hình ảnh, sau đó chuyển qua các hàm kích hoạt ReLU để tăng tính phi tuyến tính cho mạng. Quá trình này tiếp tục qua nhiều lớp CNN với sự thay đổi số lượng kênh (channels), từ đó trích xuất các đặc trưng cấp cao hơn từ dữ liệu đầu vào.



Hình 1. CNN with RL

4.2. Kiến trúc Fully Connected

Sau khi các đặc trưng không gian đã được trích xuất qua các lớp CNN, kết quả sẽ được chuyển qua các lớp Fully Connected. Các lớp này có nhiệm vụ tổng hợp và chuyển đổi các đặc trưng đã học thành các giá trị Q cho mỗi hành động có thể. Mỗi lớp FC sử dụng hàm kích hoạt ReLU, giúp tăng tính phi tuyến cho mô hình. Mạng kết thúc bằng một lớp FC với số lượng neuron tương ứng với số lượng hành động có thể.



Hình 2. Fully Connected

4.3. Forward

Khi một đầu vào được truyền qua mạng, phương thức forward sẽ thực hiện các bước sau:

- **K**iểm tra định dạng của đầu vào để đảm bảo rằng nó có ít nhất 3 chiều (bởi vì mạng này chủ yếu xử lý dữ liệu hình ảnh).
- **T**iến hành chập qua các lớp CNN để trích xuất các đặc trưng không gian.
- **K**ết quả của CNN sẽ được làm phẳng (flatten) thành một vector một chiều để đưa vào các lớp FC.
- **C**uối cùng, lớp FC sẽ xuất ra các giá trị Q cho mỗi hành động có thể, từ đó giúp xác định chính sách hành động của mạng.

5. Kết luận

Trong dự án này, chúng tôi đã phát triển và đào tạo một Agent học tăng cường (RL) bằng nền tảng MAgent2 để tham gia vào một môi trường chiến đấu cụ thể. Hiệu suất của Agent được đánh giá với ba loại đối thủ: random, tác nhân được đào tạo trước, và tác nhân được đào tạo cải tiến mạnh hơn.

Kết quả cho thấy tác nhân của chúng tôi vượt trội trước các tác nhân ngẫu nhiên, minh chứng khả năng xử lý các đối thủ chưa được đào tạo. Tuy nhiên, tác nhân gặp khó khăn khi đối mặt với các tác nhân mạnh hơn, đặc biệt là tác nhân cuối cùng được đào tạo bằng phương pháp tự chơi và Deep Q-Networks (DQN). Chúng tôi sẽ cố gắng nỗ lực để phát triển, cải thiện mô hình giúp nó hoạt động tốt hơn trong tương lai.

Tài liệu

https://magent2.farama.org/introduction/basic$_usage/$
$https://github.com/Farama-Foundation/MAgent2$
$http://incompleteideas.net/book/RLbook2020.pdf$