Báo cáo môn học Reinforcement Learning

Trần Phạm Hoàng
UET, VNU
22022669@vnu.edu.vn

Trịnh Đắc Phú
UET, VNU
22022597@vnu.edu.vn

Tạ Nguyên Dũng
UET,VNU
22022546@vnu.edu.vn

***Abstract*—Báo cáo trình bày việc áp dụng Deep Q-Networks (DQN) để huấn luyện các tác nhân trong môi trường MARL, cụ thể là battle Magent2. Mục tiêu chính là sử dụng DQN để tối ưu hóa action của các agent thông qua việc học từ các tương tác với môi trường. Để cải thiện hiệu suất, Replay Memory được sử dụng để lưu trữ và lấy mẫu các transitions trong quá khứ, trong khi mạng mục tiêu được cập nhật định kỳ để ổn định quá trình học. Các kết quả thử nghiệm trong môi trường Battle Magent2 cho thấy hiệu quả của phương pháp này, với các agents học được cách thực hiện nhiệm vụ một cách hiệu quả, có được tỉ lệ thắng 100% trước random policy và các mô hình pretrain.**

***Index Terms*—Reinforcement Learning, Battle agents, DQN, MARL**

I. Giới thiệu

A. Môi trường huấn luyện

Magent2 là một nền tảng mã nguồn mở được phát triển bởi Farama Foundation. Bài tập sẽ sử dụng môi trường Battle (battle_v4) của Magent2. Môi trường bao gồm 162 tác tử, trong đó có 81 quân đỏ và 81 quân xanh. Reward của mỗi tác tử dựa vào độ hiệu quả của tác tử đó và không phụ thuộc vào các tác tử xung quanh. Mỗi tác nhân chỉ có thể quan sát một phần nhỏ của môi trường, điều này tạo ra các yếu tố không chắc chắn và phức tạp cho việc học. Điều này yêu cầu các agents phải phối hợp chiến lược một cách hiệu quả để tối đa hóa reward.

B. Thuật toán sử dụng

Trong bài tập này thuật toán *DQN* sẽ được sử dụng để huấn luyện các agents. Deep Q-Network (DQN) là một phương pháp kết hợp giữa Học Tăng cường (Reinforcement Learning) và Mạng Nơ-ron Sâu (Deep Neural Networks) để giải quyết các bài toán với không gian trạng thái lớn, nơi việc sử dụng các phương pháp Q-learning truyền thống trở nên khó khăn.

- Quá trình huấn luyện DQN bao gồm bốn điểm chính:
  - Tạo mẫu trải nghiệm (Experience Replay): Các tác nhân lưu lại các transition $l\angle s, a, r, s, d'r\angle$ vào một bộ đệm (replay buffer) chung.
  - Huấn luyện mạng nơ-ron Q (Q-Network): Các transition được lấy từ replay buffer để cập nhật các trọng số của mạng. Mục tiêu là giảm thiểu sai số giữa giá trị $Q(s, a)$ do mạng dự đoán và mục tiêu Q-learning $r + \gamma \max_{\{a'\}} Q(s', a')$.
  - Sử dụng mạng mục tiêu (Target Network): Để ổn định việc huấn luyện, một mạng mục tiêu được sử dụng để tính toán giá trị mục tiêu, giúp tránh sự thay đổi liên tục trong quá trình huấn luyện.
  - ε-greedy: Agent sẽ chọn action ngẫu nhiên với xác suất ε và chọn action tối ưu dựa trên $Q(s, a)$ với xác suất $1 - \varepsilon$.

TABLE I: Tỉ lệ thắng của mô hình trước các đối thủ

Red agent	Blue agent	Win-rate Red	Win-rate Blue	Avg Re-ward Red	Avg Re-ward Blue
ran-dom	Model	0	1.0	−1.10	4.92
red	Model	0	1.0	0.23	4.99
red_final	Model	0	1.0	1.51	4.90

- Average score of Model for each opponent models
  Random: 1
  Pretrain-0: 1
  New Pretrain: 1

II. Phương pháp huấn luyện

Phương pháp huấn luyện của mô hình Deep Q-Network (DQN) trong môi trường Battle Magent2 được thực hiện thông qua ba thành phần chính: Mạng Q (Q-Network), Bộ nhớ kinh nghiệm (Replay Buffer) và Bộ huấn luyện (Trainer). Qua đó đảm bảo rằng các agents có thể học hỏi từ kinh nghiệm trong quá khứ và tối ưu hóa chính sách hành động của mình nhằm tối đa hóa phần thưởng tích lũy.

1) Cấu trúc mô hình DQN:
   - Mạng Q được sử dụng là một mạng CNN kết hợp với các lớp fully connected layers. Đầu vào của mạng là observation của agent. Cấu trúc của mạng bao gồm:
     - Lớp tích chập (Convolutional Layers):
       - Hai lớp tích chập liên tiếp để trích xuất các đặc trưng không gian từ đầu vào.
       - Mỗi lớp sử dụng bộ lọc có kích thước 3x3 và hàm kích hoạt ReLU.
     - Fully Connected Layers:
       - Vector đặc trưng phẳng được đưa qua ba lớp fully connected có kích thước đầu ra số hành động n_action.
       - Lớp đầu ra trả về các giá trị Q tương ứng với mỗi hành động khả thi mà tác nhân có thể thực hiện.
     - Đầu ra (Output):
       - Giá trị Q cho mỗi hành động.

2) Lớp ReplayMemory:
   - Replay Memory được triển khai dưới dạng một mảng các hàng đợi hai đầu (deque) để lưu trữ các transitions của các agent dưới dạng (trạng thái, hành động, phần thưởng, trạng thái tiếp theo, trạng thái kết thúc). Các transitions này được ghi lại trong quá trình agents tương tác với môi trường.
3) Lớp Trainer:
   - Trainer sẽ thực thi việc huấn luyện của mô hình. Policy DQN và mô hình Target DQN đều được khởi tạo với các tham số giống nhau. Mô hình Target DQN được cập nhật theo chu kỳ dựa trên sự kết hợp giữa tham số của mô hình policy DQN và mô hình target DQN với hệ số TAU.
   - Trong quá trình huấn luyện, các tác nhân thực hiện các hành động theo chính sách epsilon-greedy, trong đó xác suất chọn hành động ngẫu nhiên giảm dần theo thời gian (bắt đầu từ epsilon_start và giảm dần đến epsilon_end).
   - Sau mỗi vòng lặp, các transitions được lưu trữ vào Replay Memory. Policy DQN sẽ được tối ưu hóa bằng cách lấy các Transitions từ Replay Memory để tính toán giá trị Q hiện tại và giá trị Q mục tiêu và tối ưu hóa loss. Giá trị Q mục tiêu được tính bằng công thức:
     - $Q_{\text{tg}} = r + \gamma(1-d)\max Q_{\text{tg}}(s', a')$
       - Trong đó, r là phần thưởng,
         γ là hệ số chiết khấu,
         d = 1 là nếu s' trạng thái kết thúc và bằng 0 nếu ngược lại,
         $Q_{\text{tg}}(s', a')$ là giá trị Q của trạng thái tiếp theo
4) Quy trình huấn luyện
   - Tương tác với môi trường

Các agents tương tác với môi trường theo epsilon-greedy. Ở mỗi bước, agent quan sát trạng thái, chọn hành động, nhận phản hồi từ môi trường (phần thưởng, trạng thái tiếp theo, tín hiệu kết thúc) và tiếp tục hành động cho đến khi bị hạ hoặc kết thúc episode.

- Lưu trữ trải nghiệm

Mỗi trải nghiệm (trạng thái, hành động, phần thưởng, trạng thái tiếp theo, tín hiệu kết thúc) được lưu vào Replay Memory.

- Lấy mẫu và huấn luyện Q-Network

Các mẫu từ Replay Memory được DataLoader chia thành batch ngẫu nhiên. Policy Q-Network dự đoán giá trị Q và so sánh với giá trị Q mục tiêu, tính toán MSE Loss và tối ưu tham số mô hình dựa trên Loss.

- Cập nhật Target Q-Network

Target Q-Network được cập nhật định kỳ để giảm biến động trong quá trình huấn luyện. Tham số của Target Q-Network là tổ hợp tuyến tính của tham số cũ và tham số của Policy DQN, giúp tăng độ ổn định cho quá trình huấn luyện.

III. KẾT QUẢ

Chúng tôi thực hiện tinh chỉnh tham số và tìm ra được bộ tham số mà chúng tôi thấy rằng là mạnh nhất, với những tham số này chúng tôi tiếp tục thay đổi step_reward của môi trường vì chúng tôi nhận thấy rằng khi game đấu đi về giai đoạn cuối các agent sẽ chỉ hoạt động ở quanh khu vực quan sát được của nó mà không đi tìm kiếm đối thủ. Chúng tôi tăng điểm phạt cho step_reward để mô hình của chúng tôi sẽ thực hiện việc tìm kiếm và chủ động tấn công đối thủ. Sau quá trình này chúng tôi có được 3 mô hình lần lược là DQN_v1(với step_reward là mặc định), DQN_v2(step_reward=-0.01), DQN_v3(step_reward=-0.05).

TABLE II: Comparison of Agents and Their Win Rates

Red agent	Blue agent	Win-rate Red	Win-rate Blue	Av Re Red	Av Re Blue	Game Steps
ran-dom	DQN_v1	0	1.0	−2.13	4.48	16492
red	DQN_v1	0	1.0	1.10	4.15	17253
red_final	DQN_v1	0	1.0	−1.25	1.27	37253
ran-dom	DQN_v2	0	1.0	−1.52	4.66	11491
red	DQN_v2	0	1.0	2.86	4.82	7409
red_final	DQN_v2	0	1.0	2.5	3.86	19878
ran-dom	DQN_v3	0	1.0	−1.10	4.92	6198
red	DQN_v3	0	1.0	0.23	4.99	5375
red_final	DQN_v3	0	1.0	1.51	4.90	6295

Để đánh giá hiệu suất của mô hình chung tôi cho mô hình đấu với 3 mô hình khác nhau (random, red.pt, red_final.pt). Từng mô hình trên sẽ đấu 100 game với mô hình của chúng tôi. Với mỗi game sẽ tính phần thưởng trung bình của mỗi agent và số ván thắng của từng bên rồi tính trung bình của 100 game, một bên thằng là khi hết game mà số lượng agent của bên thắng hơn bên thua từ 5 trở lên nếu không ván đó sẽ tính là hoà.

Có thể thấy trong Table II các mô hình DQN của chúng tôi có tỉ lệ thắng so với các mô hình của red Agent là 100% và có thể thấy khi để step_rewared phạt cao mô hình học được cách chủ động đi tìm và tấn công đối thủ thay vì chờ đợi đối thủ đến gần rồi mới tấn công

IV. Discussion

Trong quá trình huấn luyện và đánh giá các mô hình DQN của chúng tôi (DQN_v1, DQN_v2, DQN_v3), kết quả cho thấy sự ảnh hưởng rõ rệt của các tham số trong quá trình

huấn luyện, đặc biệt là tham số step_reward, đến hiệu suất của các tác nhân. Cụ thể, chúng tôi nhận thấy rằng khi giảm giá trị của step_reward (ví dụ: −0.01, −0.05), mô hình đã thể hiện hành vi chủ động hơn, tích cực tìm kiếm và tấn công đối thủ, thay vì chỉ phản ứng khi đối thủ tiếp cận.

Một điểm đáng chú ý là tất cả các mô hình DQN của chúng tôi đều có tỉ lệ thắng 100% so với các mô hình red agent, cho thấy khả năng học hỏi và cải thiện của mô hình qua quá trình huấn luyện. Các tác nhân DQN đã học được cách tối ưu hóa hành động của mình thông qua việc điều chỉnh Q-values, nhờ đó chúng có thể đưa ra các quyết định chính xác hơn trong môi trường Battle phức tạp này.

Sự thay đổi step_reward đã chứng minh ảnh hưởng sâu sắc đến chiến lược của tác nhân. Ban đầu, khi step_reward có giá trị gần 0 (như trong DQN_v1), các tác nhân chỉ tập trung vào việc bảo vệ và không chủ động tấn công. Tuy nhiên, khi giá trị step_reward bị phạt mạnh hơn (DQN_v2 và DQN_v3), các tác nhân đã học được cách di chuyển chủ động và tìm kiếm đối thủ để tấn công, điều này giúp cải thiện đáng kể hiệu suất của chúng trong môi trường.

Ngoài ra, việc sử dụng các mô hình Replay Memory và epsilon-greedy giúp cho quá trình huấn luyện diễn ra ổn định và hiệu quả. Replay Memory giúp tác nhân không bị phụ thuộc vào một loạt các trải nghiệm gần đây, giúp quá trình học của tác nhân trở nên linh hoạt và tổng quát hơn. Chính sách epsilon-greedy giúp cân bằng giữa việc khai thác (exploitation) các chiến lược đã học và khám phá (exploration) những hành động mới, từ đó giúp tác nhân không bị mắc kẹt trong những chiến lược địa phương không tối ưu.

Cuối cùng, các kết quả từ bảng đánh giá Table II cho thấy rằng việc tinh chỉnh tham số huấn luyện, đặc biệt là giá trị của step_reward, đã mang lại những cải thiện rõ rệt về mặt hiệu suất. Các mô hình DQN_v2 và DQN_v3 đã thể hiện sự vượt trội trong các trận đấu, điều này cho thấy rằng việc điều chỉnh tham số môi trường có thể ảnh hưởng sâu sắc đến cách thức tác nhân học và ra quyết định trong các tình huống thực tế.

V. Conclusion

Báo cáo trên đã trình bày quy trình huấn luyện các tác nhân trong môi trường Battle Magent2 bằng thuật toán Deep Q-Network (DQN). Bằng cách sử dụng Replay Memory để lưu trữ toàn bộ trải nghiệm và huấn luyện trên tất cả các mẫu với DataLoader, mô hình đạt được khả năng học tập ổn định và hiệu quả hơn. Việc cập nhật Target Q-Network theo phương pháp “soft update” giúp giảm thiểu dao động trong quá trình huấn luyện. Trong tương lai, việc mở rộng mô hình hoặc tích hợp các thuật toán khác(như là Q-mix hoặc PPO) có thể là những mở rộng tiềm năng để cải thiện hiệu suất của agents.

References