Báo cáo bài tập lớn
Môn học: Học tăng cường và lập kế hoạch
Nhóm 34

Bùi Quang Vinh
MSSV: 22022529
link github: https://github.com/VinhLL/RL-final-project-AIT-3007.git

***Tóm tắt*—Báo cáo được thực hiện bởi nhóm 34, thực hiện giải quyết bài toán huấn luyện một mô hình học tăng cường để tiến hành đối kháng với tác tử đã được huấn luyện từ trước được đưa ra từ giảng viên trong môi trường MAgent2.**

I. Giới thiệu

Phương pháp tiếp cận: Cải tiến QNetwork có sẵn bằng cách giảm số kênh trong các lớp convolution xuống còn 8. Mạng mới sử dụng padding và stride để điều chỉnh quá trình trích xuất đặc trưng. Thêm 1 lớp dropout với tỉ lệ 0.2 vào sau lớp fully connected để có thể giảm overfitting. Để có thể giúp mạng học được các đặc trưng phức tạp hơn, số đơn vị trong các lớp fully connected cũng được thay đổi từ 120 lên thành 128.

TABLE I
Tóm tắt kết quả

Model	Win	Draw	Lose
Random	100 ± 0	0 ± 0	0 ± 0
Pretrain-0	100 ± 0	0 ± 0	0 ± 0
New Pretrain	100 ± 0	0 ± 0	0 ± 0

Điểm trung bình cho mỗi mô hình, sử dụng 1 điểm cho thắng, 0.5 điểm cho hòa và 0 điểm cho thua:

- Random: 100
- Pretrain-0: 100
- New Pretrain: 100

II. Phương pháp

Phương pháp chính sử dụng ở đây là dùng Deep-Q-Network (DQN) để dự đoán $Q - valuefunction$ tương ứng với mỗi cặp hành động a và trạng thái s. Mục tiêu của mạng là sử dụng Bellman equation để cập nhật giá trị $Q(s, a)$:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t) \right]$$

Trong đó:

- $Q(s_t, a_t)$: giá trị Q của trạng thái s_t và hành động a_t
- α: learning rate
- γ: hệ số chiết khấu ($0 \leq \gamma \leq 1$)
- $\max_{a'} Q(s_{t+1}, a')$: giá trị Q tối ưu ở trạng thái tiếp theo s_{t+1}, đạt kỳ vọng lớn nhất khi thực hiện hành động tối ưu a'
- r_{t+1}: Phần thưởng khi thực hiện hành động a_t tại trạng thái s_t

DQN sử dụng Replay Buffer và Target Network để có thể ổn định quá trình huấn luyện:

- Replay Buffer: Bộ nhớ đệm lưu trữ các cặp chuyển tiếp s, a, r, s' cho phép lấy mẫu ngẫu nhiên giúp giảm sự tương quan giữa các mẫu liên tiếp.
- Target Network: Mạng riêng biệt được cập nhật chậm để tính toán các giá trị ổn định.

III. Implementation

Phần thực hiện được tiến hành bằng cách sử dụng GPU T4 x2. Việc train kéo dài trong 12 phút với 100 episode trên Kaggle.

A. Môi trường

Ở đây, mô hình được chạy trên môi trường battle_v4 nằm trong thư viện magent2. Thông số của môi trường:

- Map size: 45x45
- Số bước tối đa mỗi episode: 300
- Phần thưởng:
  - step_reward: 0.005
  - dead_penalty: 0.1
  - attack_penalty: 0.05
  - attack_opponent_reward: 0.5

B. Kiến trúc Neural Network

Mỗi Q-Network được cấu tạo gồm các thành phần chính:

- Convolutional Layers: 2 tầng tích chập với kích thước kernel 3x3, số filter là 8 và kích hoạt bởi hàm ReLU.
- Fully Connected Layers: Tầng thứ nhất gồm 128 đơn vị, hàm kích hoạt là ReLU. Sử dụng Dropout 20% để giảm overfitting. Tầng thứ hai có đầu ra ứng với không gian hành động.
- Output Layer: Tầng tuyến tính được kết nối trực tiếp với số action.

C. Quy trình huấn luyện

Quy trình:

- Replay Buffer:
  - Capacity: 60,000
  - Lưu trữ (state, action, reward, next_state, done)

- Hyperparameters:
  - Batch Size: 1024
  - Discount Factor (γ): 0.9
  - Optimizer: Adam với learning rate 1×10^{-3}, giảm dần sau mỗi 3 episode
  - Tần suất cập nhật: 3 episode

Training loop:

- epsilon-greedy policy:
  - Chọn ngẫu nhiên với xác suất ϵ.
  - Nếu ϵ nhỏ, chọn hành động tối ưu từ DQN.
- Cập nhật replay buffer: Sau mỗi bước, lưu trữ thông tin vào replay buffer.
- Huấn luyện DQN:
  - Loss function sử dụng MSE giữa Q-value hiện tại và giá trị mục tiêu: $Q_{target} = r + \gamma \cdot Q_{next} \cdot (1 - d)$
  - Gradient sử dụng Adam để cập nhật.
- Cập nhật mạng mục tiêu từ mạng chính.
- Giảm ϵ theo công thức $\epsilon = \max(\epsilon \cdot \epsilon_{decay}, \epsilon_{min})$.

IV. KẾT QUẢ VÀ ĐÁNH GIÁ

A. Kết quả

Kết quả huấn luyện: Kết quả thực hiện đấu với các agent

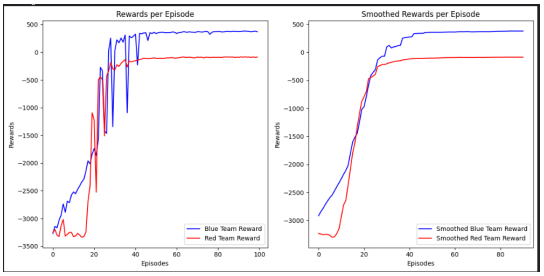

Fig. 1. Rewards per Episode

cho sẵn:

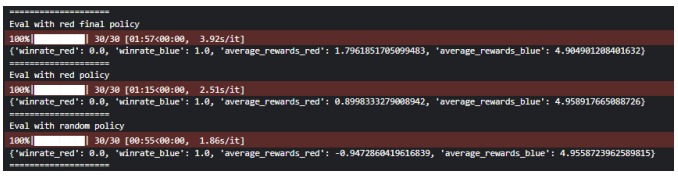

Fig. 2. Enter Caption

Thời gian huấn luyện: 12 phút

Nhận xét: Về kết quả huấn luyện:

- Trong quá trình huấn luyện, ban đầu reward của blue rất thấp, chưa tối ưu.
- ϵ giảm dần từ 1.0 đến 0.1 giúp chuyển từ chọn ngẫu nhiên sang học theo chính sách.
- Sau khoảng 25 episode, reward dần ổn định ở mức dương, mô hình dần hội tụ và học được policy tối ưu, từ đó reward của blue cao hơn reward của red.

Về cách chọn Hyperparameter:

- $\gamma = 0.9$ phù hợp cho môi trường quan trọng dài hạn nhưng tương lai không nên quá ảnh hưởng.
- ϵ và ϵ Decay giảm đều từ 1 và tối thiểu là 0.1 đảm bảo việc chuyển giai đoạn được mượt mà.
- Replay Buffer với Capacity: 60,000 được tính toán đủ lớn, với batch_size 1024 không quá lớn để đảm bảo bộ nhớ, được thử lại với nhiều giá trị để tìm ra tham số tối ưu.

V. CONCLUSION

Việc áp dụng phương pháp Deep Q-Network (DQN) để huấn luyện mô hình trong môi trường magent2 với các cải tiến so với mạng DQN ban đầu đã đạt được hiệu suất cao với tỷ lệ thắng đối với cả 3 mô hình là 100%. Replay Buffer và Target Network được sử dụng giúp cho việc huấn luyện đạt được sự ổn định, khi kết hợp với epsilon-greedy đã giúp cho việc chuyển từ khám phá sang khai thác một cách hoàn chỉnh. Thời gian huấn luyện chỉ 12 phút cho 100 tập cũng cho thấy việc chọn Hyperparameter là hợp lý.

REFERENCES

[1] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.
[2] MAgent2. Link: https://github.com/Farama-Foundation/MAgent2