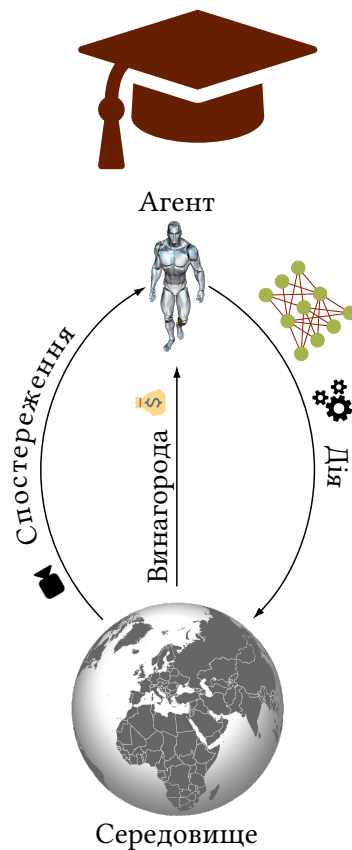




Навчання підкріпленням

Методичні вказівки для виконання практичних робіт | Осінній семестр



Глибинне Q-навчання

“Якщо Ви зробили помилку, це означає, що Ви пробували щось нове.”
– Альберт Ейнштейн

Виконуючи це завдання, Ви познайомитеся з тим як здійснювати передбачення найкращих дій для заданих станів агента на основі нейронних мереж.

Вступ

Метод глибинного Q-навчання дозволив уперше досягти надлюдської продуктивності в іграх Atari [1], де представлено першу модель для успішного знаходження оптимальної стратегії безпосередньо з високорозмірних піксельних вхідних даних за допомогою навчання з підкріпленням. Відноситься глибинне Q-навчання до алгоритмів без стратегії (off-policy). Тобто глибинне Q-навчання дозволяє знайти оптимальну стратегію на основі вивченої Q-функції, а не вивчає безпосередньо саму стратегію. Іншими словами, ідея Q-навчання полягає у тому, щоб вивчити функцію цінності дій, яку часто позначають як $Q(S_t, A_t)$, де S_t – поточний стан агента у момент часу t , A_t – дія, що виконується агентом у цьому стані. Коли вивчена оптимальна Q-функція ми отримуємо також оптимальну стратегію, шляхом обрання для кожного стану максимальне значення з Q-таблиці.

Q-навчання – це форма методу часових різниць (TD-навчання), де, на відміну від методів Монте-Карло, агент може вчитися на кожному кроці, а не чекати завершення епізоду. Ідея полягає в тому, що як тільки агент виконає дію він переходить в новий стан, далі він використовує поточне значення $Q(S_t, A_t)$ цього стану як оцінку майбутніх винагород $Q(S_{t+1}, a)$.

$$\underset{\text{Нове значення}}{Q_{k+1}(S_t, A_t)} \leftarrow Q_k(S_t, A_t) + \alpha_t [R_{t+1} + \gamma \max_a Q_k(S_{t+1}, a) - Q_k(S_t, A_t)] \quad (1)$$

де α_t – швидкість навчання, γ – коефіцієнт знецінювання, R_{t+1} – отримана агентом винагорода.

Коли простір дій та станів агента є неперервним ми не можемо просто використовувати табличне представлення Q-функції, натомість, замість дискретизації цих просторів часто використовують апроксиматор функції. Ідея апроксимації функції полягає у введенні нового параметра θ , який дозволяє вивчити оптимальну Q-функцію $Q^*(s, a)$:

$$Q^*(s, a) \approx \hat{Q}(s, a, \theta) \quad (2)$$

де θ – ваги мережі.

Таким чином ми отримуємо задачу навчання з учителем, де \hat{Q} є наближення, а $R + \gamma \max_a Q(s', a)$ – цільовим значенням. Потім ми використовуємо середньоквадратичну помилку як функцію втрат і відповідно оновлюємо вагові коефіцієнти за допомогою градієнтного спуску. У якості апроксиматора функцій використовують нейронні мережі. Для отримання більшої інтуїції щодо глибинного Q-навчання від одного з авторів [1], перегляньте наступне відео: <https://www.youtube.com/watch?v=fevMOp5TDQs>

Завдання для виконання

Відкрийте цей приклад в Colab.

- Дослідіть середовище LunarLander-v3 та поведінку ненавченого агента.
- Ознайомтесь з реалізацією алгоритму DQN.
- Реалізуйте алгоритм Double DQN (DDQN).
- Реалізуйте Dueling Deep Q-Networks (Dueling DQN).
- Навчіть агента.
- Порівняйте їхню ефективність для DQN, DDQN та Dueling DQN.
- Продемонструйте поведінку навченого агента для різних алгоритмів.
- Зробіть власні висновки та спостереження.

Приклади реалізації DDQN і Dueling DQN можна знайти тут: <https://www.kaggle.com/code/auxeno/dqn-on-lunar-lander-rl>

Оцінювання

Максимальна оцінка за виконання завдання – 10 балів.

Здача завдання

Відправляєте на перевірку СЮДИ: <https://cloud.comsys.kpi.ua/s/7xQfSxdZg83MXRA>:

детально прокоментовану програмну реалізацію завдання, порівняння ефективності реалізованих алгоритмів, висновки та спостереження: Прізвище Ім'я_група_DQN.ipynb

Дедлайн: 10 грудня 2025 року о 23:59

Література

- [1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “Human-level control through deep reinforcement learning,” *nature*, vol. 518, no. 7540, pp. 529–533, 2015. [Online]. Available: <https://daiwk.github.io/assets/dqn.pdf>