Politechnika Wrocławska

Algorytmy sztucznej inteligencji w Przemyśle 4.0

Uczenie ze wzmocnieniem

Dr inż. Radosław Idzikowski

HR EXCELLENCE IN RESEARCH

Uczenie maszynowe

Uczenie maszynowe (*machine learning*)

Obszar sztucznej inteligencji poświęcony algorytmom, które poprawiają się automatycznie poprzez doświadczenie, czyli ekspozycję na dane[a]

[a] wikipedia

Uczenie maszynowe

- uczenie nadzorowane:
  - klasyfikacja,
  - predykcja.
- uczenie nienadzorowane:
  - grupowanie,
  - redukcja wymiarów,
  - uzupełnianie wartości.
- uczenie ze wzmocnieniem.

Metody uczenia ze wzmocnieniem

Uczenie pasywne:

- ocena strategii (*Policy Evaluation*),
- polepszanie strategii (*Policy Improvement*),
- iteracyjne doskonalenie strategii (*Policy Iteration*),
- iteracyjne obliczanie funkcji wartości (*Value Iteration*).

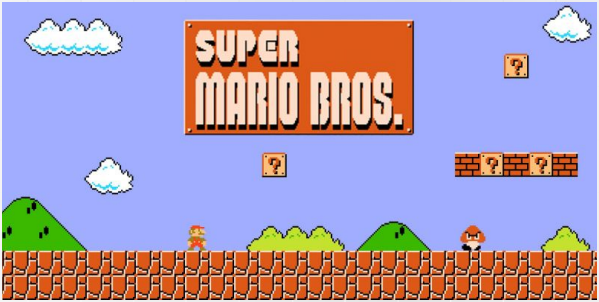
Uczenie aktywne:

- metody różnic czasowych (*Temporal Difference Learning*):
  - Monte Carlo,
  - Q-Learning
  - SARSA
- metody aproksymacyjne,
- metody wykorzystujące głębokie sieci neuronowe:
  - Deep Q-Learning,
  - Double Q-Learning,
  - Actor-Critic,
  - REINFORCE
  - Policy Gradient.

Uczenie ze wzmocnieniem a inne metody uczenia maszynowego

- Nie jest potrzebny zbiór danych.
- Nagroda może być odłożona w czasie.
- Czas ma znaczenie.
- Działanie agenta ma wpływ na dane, jakie otrzymuje ze środowiska.

Super Mario Bros



Rysunek 1: Super Mario Bros

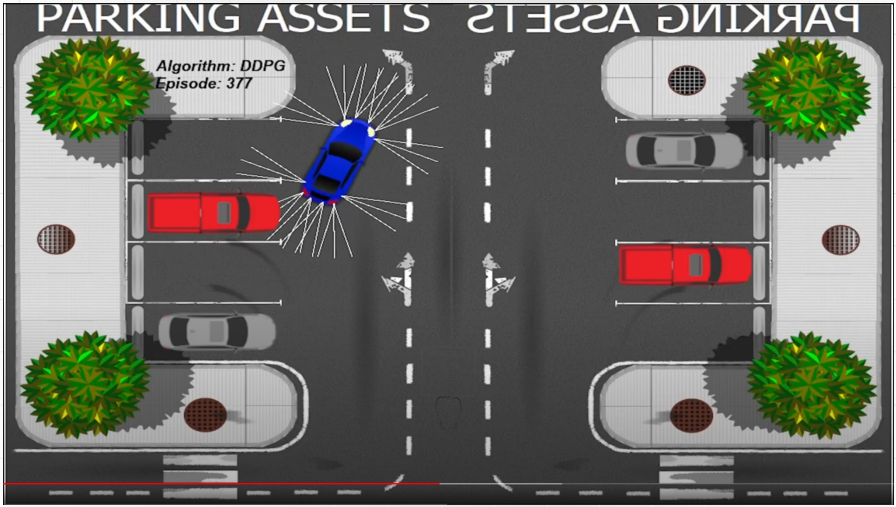
- Tsay, Jyh-Jong, Chao-Cheng Chen, and Jyh-Jung Hsu. "Evolving intelligent mario controller by reinforcement learning." 2011 International Conference on Technologies and Applications of Artificial Intelligence. IEEE, 2011.
- Shu, Tianye, Jialin Liu, and Georgios N. Yannakakis. "Experience-driven PCG via reinforcement learning: A Super Mario Bros study." 2021 IEEE Conference on Games (CoG). IEEE, 2021.

Pięć zasad

Podczas projektowania metod bazujących na uczeniu ze wzmocnieniem należy kierować się następujhącymi zasadami:

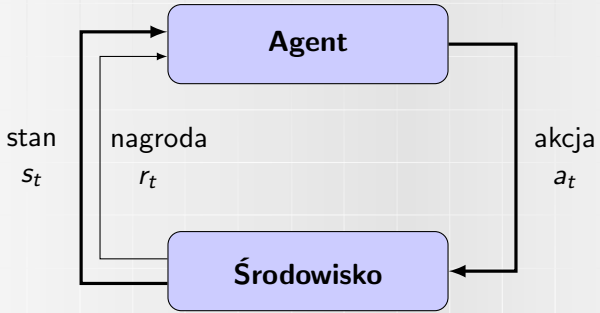
1. system wejścia i wyjścia,
2. nagroda,
3. środowisko,
4. proces decyzyjny Markowa,
5. szkolenie i wnioskowe.

Zastosowanie



Rysunek 2: *Automatyczne parkowanie z wykorzystaniem metod sztucznej inteligencji* – praca magisterska A. Sobecki (2023)

Uczenie ze wzmocnieniem



Rysunek 3: Schematyczne przedstawienie podstawowych pojęć w ramach RL: Agent wchodzi w interakcję z środowiskiem poprzez podjęcie akcji A_t, ta natomiast prowadzi do zmiany stanu środowiska na S_t oraz otrzymaniem nagrody lub kary R_t przez agenta za podjętą akcję – proces ten następnie powtarza się w kolejnych iteracjach.

Uczenie ze wzmocnieniem

Podstawy

Środowisko (*Environment*)

To zadanie/symulacja, z którym agent wchodzi w interakcję.

- wejście:
  - akcja,
- wyjście:
  - stan,
  - nagroda

Uczenie ze wzmocnieniem

Podstawy

Agent (*Agent*)

Poprzez interakcje ze środowiskiem uczy się, jak najkorzystniejszego oddziaływania ze środowiskiem. Funkcję odpowiedzialną za wybranie odpowiniedj akcji przez agenta nazywamy polityką (*Policy*)

- wejście:
  - stan,
  - nagroda
- wyjście:
  - akcja,

Uczenie ze wzmocnieniem

Podstawy

Nagroda (*Reward*)

Wartość zwracana przez środowisko po wykonaniu akcji wybranej przez agenta.

$$r_t \tag{1}$$

Akcja (*Action*)

Działanie agenta na środowisko. Akcja może być wybrana ze zbioru lub określą wartością albo wektorem wartości.

$$a_t \tag{2}$$

Stan (*State*)

Zmienna określająca stan środowiska.

$$s_t \tag{3}$$

Uczenie ze wzmocnieniem

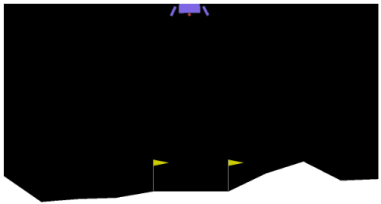
Proces decyzyjny Markowa

Proces decyzyjny Markowa (*Markov Decision Process*, MDP)

Ciąg zdarzeń, w którym prawdopodobieństwo każdego zdarzenia zależy jedynie od wyniku poprzedniego. W ujęciu matematycznym, procesy Markowa to takie procesy stochastyczne, które spełniają własność Markowa, że warunkowe rozkłady prawdopodobieństwa przyszłych stanów procesu są zdeterminowane wyłącznie przez jego biedacy stan, bez względu na przeszłość. [a]

[a] wikipedia

Gym is a standard API for reinforcement learning, and a diverse collection of reference environments



The Gym interface is simple, pythonic, and capable of representing general RL problems:

```
import gym
env = gym.make("LunarLander-v2", render_mode="human")
observation, info = env.reset(seed=42)
for _ in range(1000):
   action = policy(observation)  # User-defined policy function
   observation, reward, terminated, truncated, info = env.step(action)

   if terminated or truncated:
      observation, info = env.reset()
env.close()
```

Narzędzia

- `Google Colab` – darmowe środowisko (choć bez gwarancji dostępności zasobów) w chmurze, pozwala wykonywać kod `PYTHON` oraz jednocześnie dokumentować ten proces w ramach `Jupyter notebooks`.
- `Gym` – biblioteka zawierająca ustandaryzowaną implementację prostych, podstawowych środowisk dla RL.
- `Stable-Baselines3` – biblioteka zawierająca implementację algorytmów odpowiedzialnych za trening i decyzje agenta.
- `Anaconda` – zalecany menadżer środowiska programistycznego (nie w znaczeniu środowiska w ramach RL) dla języka `PYTHON` do zarządzania bibliotekami.

Stable-Baselines3

- `A2C` (*Advantage Actor Critic*) – łączy aktora i krytyka w jednym modelu. Wykorzystuje funkcję przewagi, aby określić, jak dobre są wybrane działania w stosunku do przewidywanej nagrody.
- `DDPG` (*Deep Deterministic Policy Gradient*) – jednocześnie uczy się funkcji Q i polityki. Wykorzystuje dane spoza polityki i równanie Bellmana do uczenia się funkcji Q, a wykorzystuje funkcję Q do uczenia się polityki.
- `DQN` (*Deep Q-learning*) – łączy uczenie Q-learning z głębokim uczeniem maszynowym. Głównym celem DQN jest szkolenie sieci neuronowej, aby funkcjonowała jako aproksymator funkcji wartości Q.
- `PPO` (*Proximal Policy Optimization*) – optymalizuje strategię agenta w sposób bezpieczny, unikając zbyt dużych zmian w polityce.
- `SAC` (*Soft Actor-Critic*) – dąży do maksymalizacji entropii polityki, zachowując losowość w wyborze działań, co sprzyja lepszej eksploracji środowiska.
- `TD3` (*Twin Delayed Deep Deterministic Policy Gradients*) – wykorzystuje deterministyczne podejście Actor-Critic z dwoma krytykami dla dokładniejszej oceny wartości stanów i akcji.

Parametry

- `learning_rate` – The learning rate, it can be a function of the current progress remaining (from 1 to 0).
- `vf_coef` – Value function coefficient for the loss calculation.
- `ent_coef` – Entropy coefficient for the loss calculation.
- ...

Frame Title

Lunar Lander