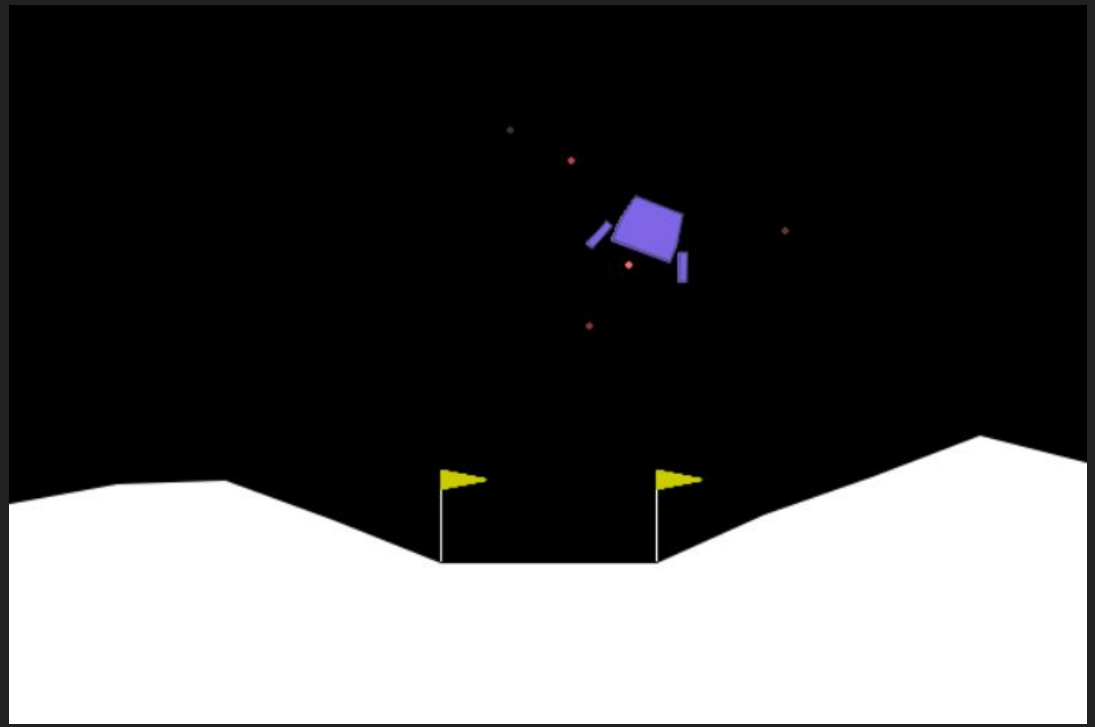
Q-learning i deep Q-learning na przykładzie Lunar Lander

Krzysztof Surówka

Lunar Lander



Środowisko i języki programowania

- Środowisko to Lunar Lander z biblioteki gymansium (pochodna biblioteki gym od OpenAI).
- Język programowania to Python.
- Do sieci neuronowych została użyta biblioteka PyTorch.
- Dodatkowo do obliczeń numerycznych została użyta biblioteka NumPy.

Stan w Lunar Lander

Stan składa się z:

- Pozycji x oraz y.
- Prędkości x oraz y.
- Kąta względem pionu i prędkości kątowej.
- Wartość 0/1 mówiącej czy lewa/prawa noga dotyka ziemi.

Łącznie daje to 8 parametrów.

Akcje w Lunar Lander

Są 4 możliwe akcje do wykonania w Lunar Lander:

- Uruchomienie głównego silnika.
- Uruchomienie prawego/lewego silnia.
- Brak uruchomienia silnika.

Silnik zostaje uruchomiony na pełnej mocy albo wcale. Istnieje również ciągła wersja tego problemu, gdzie możemy sterować również mocą silnika.

Nagrody w Lunar Lander

Nagrody w Lunar Lander są liczone w następujący sposób:

- Jest zwiększana/zmniejszana im bliżej/dalej lądownik jest od platformy.
- Jest zwiększana/zmniejszana im wolniej/szybciej porusza się lądownik.
- Jest zmniejszana w miarę większego nachylenia lądownika.
- Jest zwiększana o 10 punktów za każdą nogę, która jest w kontakcie z ziemią.
- Jest zmniejszana o 0,03 punkta za każdą klatkę, w której silnik boczny jest aktywowany.
- Jest zmniejszana o 0,3 punkta za każdą klatkę, w której silnik główny jest aktywowany.

Dodatkowo lądownik otrzymuje nagrodę -100 lub +100 punktów za zderzenie lub bezpieczne lądowanie.

Q-learning i deep Q-learning

W Q-learningu zostało użyte równanie:

$$Q(s, a) = (1 - \alpha) * Q(s, a) + \alpha * [R(s, a) + \gamma * \max(Q(s', a'))]$$

Natomiast w deep Q-learningu zostało użyte równanie Bellmana:

$$Q(s, a) = R(s, a) + \gamma * \max(Q(s', a'))$$

Dyskretyzacja stanów

Aby zastosować Q-learning, dziedzinę stanów należy zdyskretyzować.

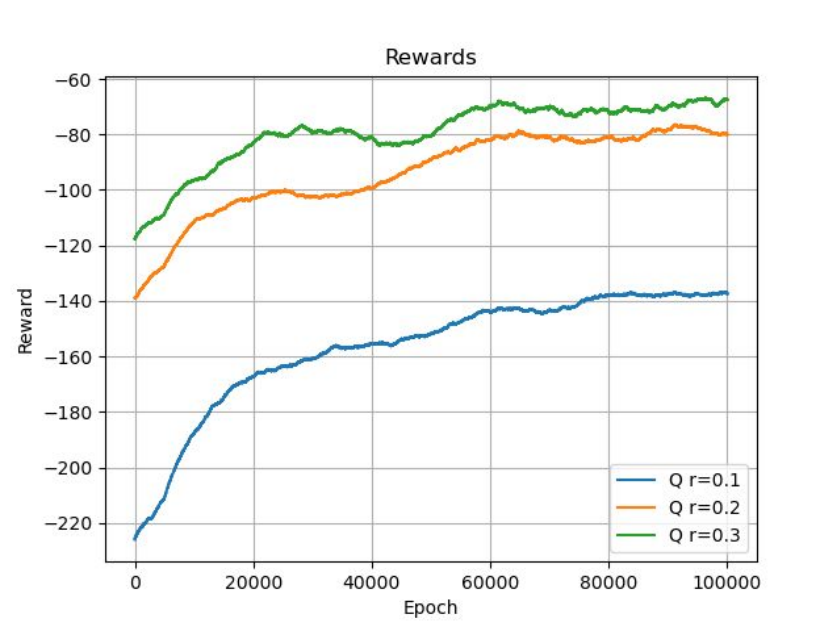
- Im większa dyskretyzacja, tym model będzie się szybciej uczył, ale może potem okazać się że będzie mało dokładny.
- Dodatkowo pojawia się problem wielkości Q-table. Im na mniejsze przedziały podzielimy dziedzinę, tym model będzie więcej zajmował miejsca.

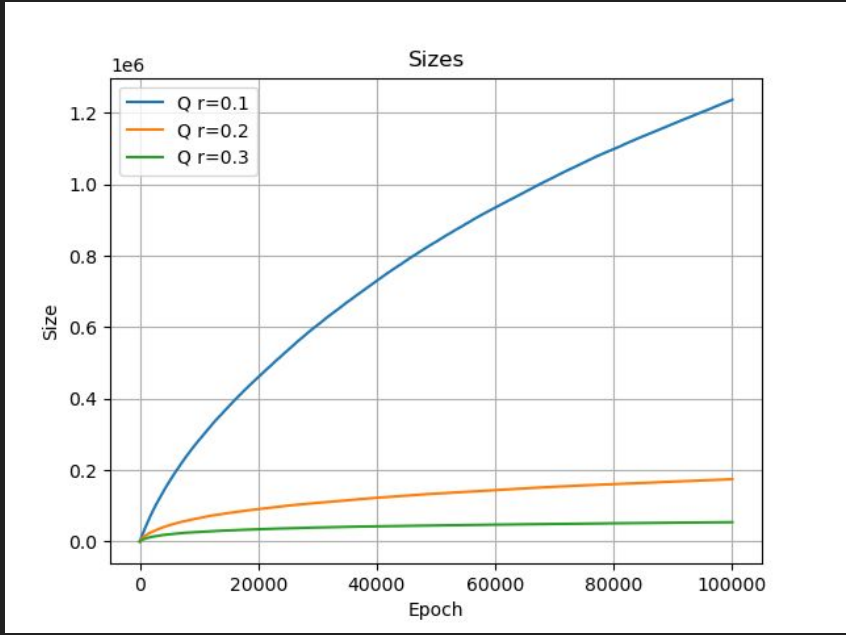
W deep Q-learning nie trzeba dyskretyzować dziedziny, bo nie mam Q-table, a sieć neuronową

Inne parametry

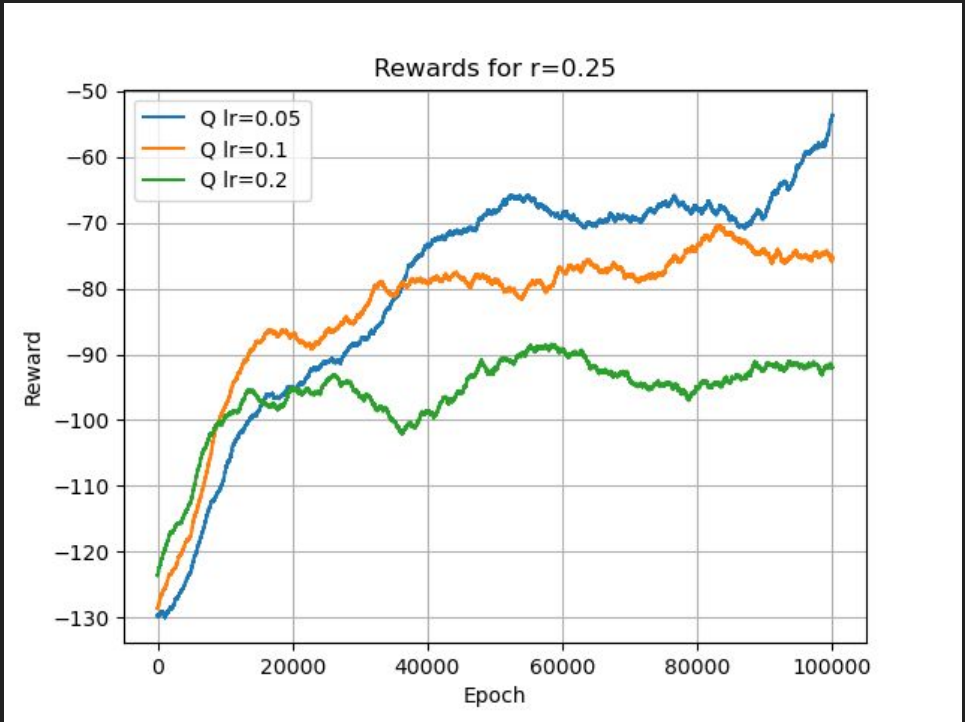
- Parametr r (podział dziedziny na kubełki o rozmiarze r)
- Parametr lr (learning rate, parametr alpha)
- Parametr y (discount dla przyszłego stanu, gamma)
- Parametr u (kubełki nie są co r, a są co r^u)
- Parametr d (losowość podejmowania akcji co epokę jest mnożona przed d, aż do osiągnięcia poziomu 0.05)
- Parametr n (wielkość warstwy ukrytej w sieci neuronowej)

Parametr r

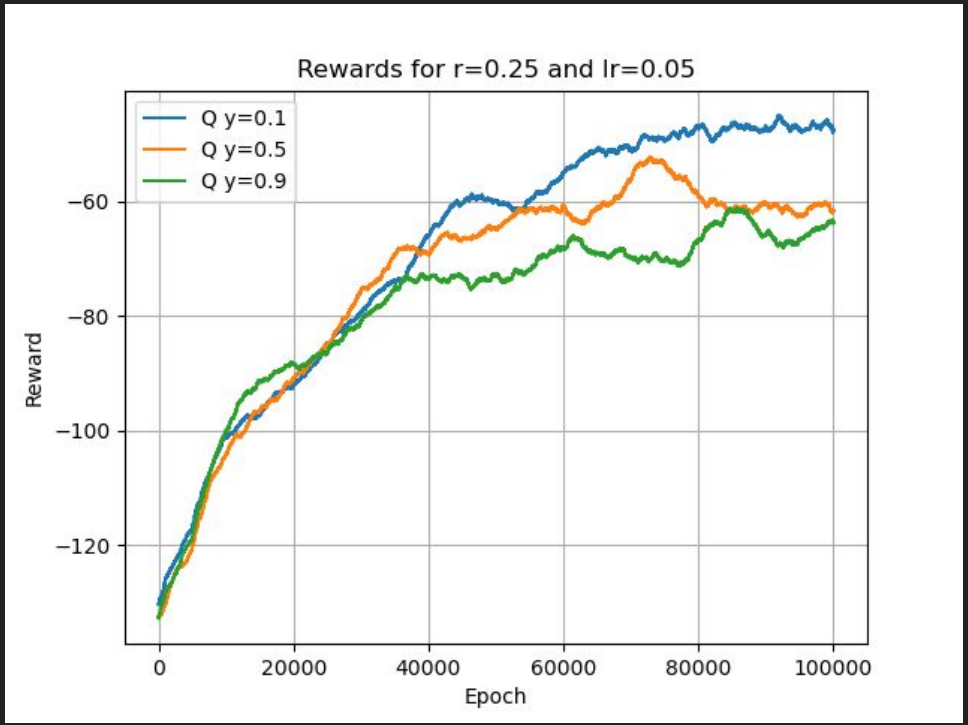

Rewards
Reward
Epoch
−60
−80
−100
−120
−140
−160
−180
−200
−220
0
20000
40000
60000
80000
100000
Q r=0.1
Q r=0.2
Q r=0.3


Sizes
1e6
Size
Epoch
1.2
1.0
0.8
0.6
0.4
0.2
0.0
0
20000
40000
60000
80000
100000
Q r=0.1
Q r=0.2
Q r=0.3

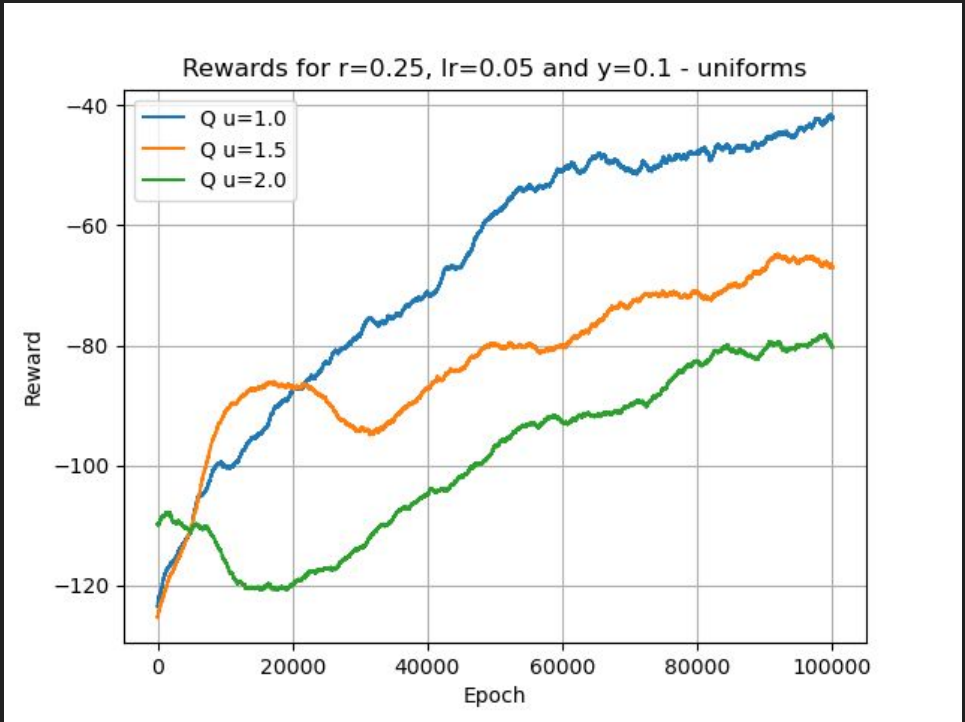
Parametr lr


Rewards for r=0.25
Q lr=0.05
Q lr=0.1
Q lr=0.2
Reward
−50
−60
−70
−80
−90
−100
−110
−120
−130
0
20000
40000
60000
80000
100000
Epoch

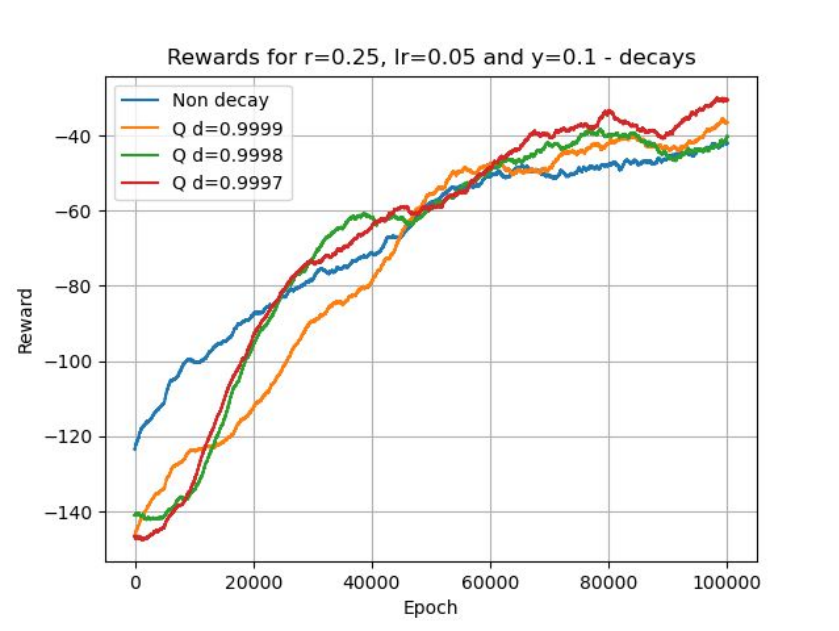
Parametr y

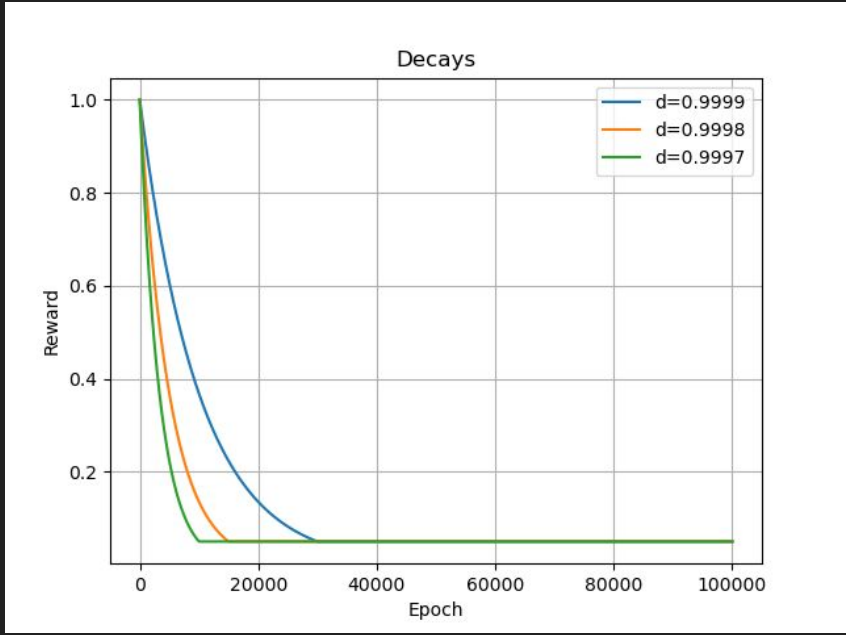

Rewards for r=0.25 and lr=0.05
Q y=0.1
Q y=0.5
Q y=0.9
−60
−80
−100
−120
Reward
0
20000
40000
60000
80000
100000
Epoch

Parametr u

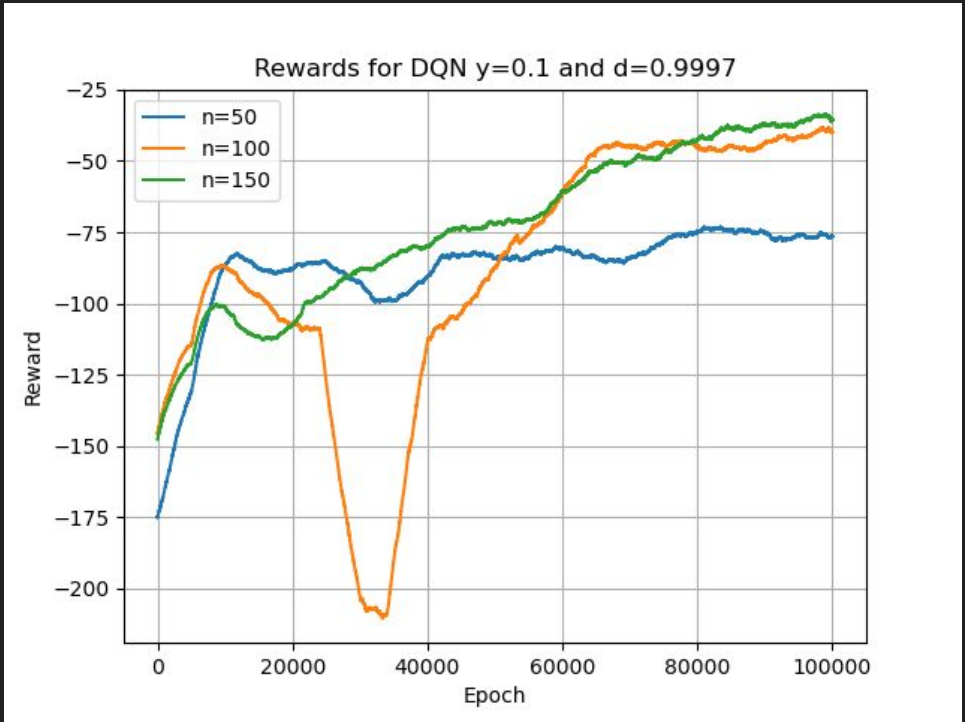

Rewards for r=0.25, lr=0.05 and y=0.1 - uniforms
Q u=1.0
Q u=1.5
Q u=2.0
−40
−60
−80
−100
−120
Reward
0
20000
40000
60000
80000
100000
Epoch

Parametr d

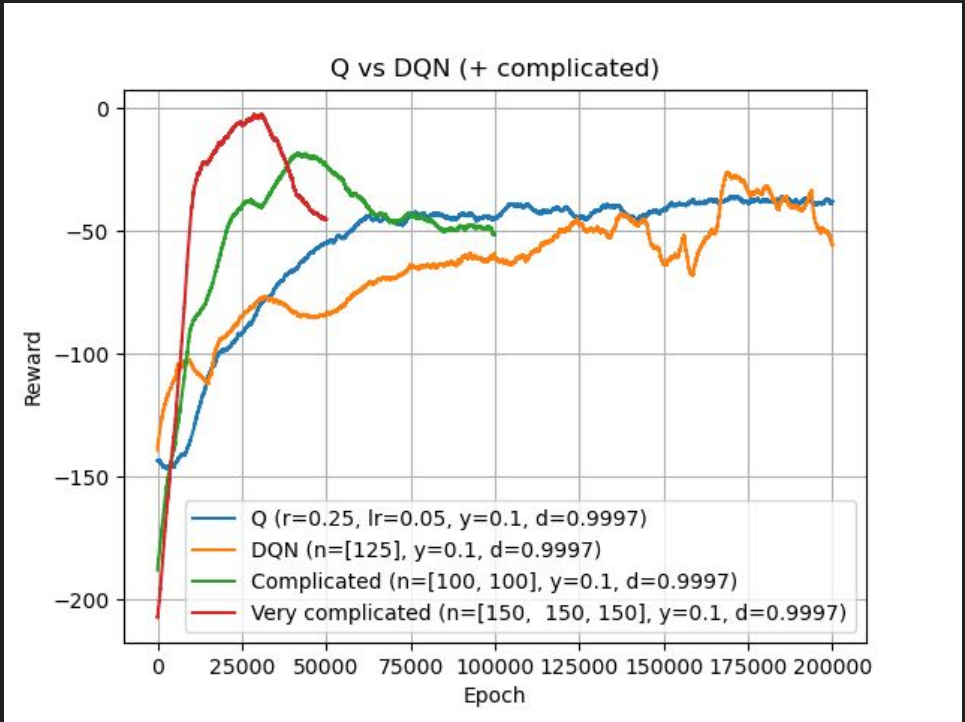

Rewards for r=0.25, lr=0.05 and y=0.1 - decays
Non decay
Q d=0.9999
Q d=0.9998
Q d=0.9997
Reward
Epoch
−40
−60
−80
−100
−120
−140
0
20000
40000
60000
80000
100000


Decays
d=0.9999
d=0.9998
d=0.9997
Reward
Epoch
1.0
0.8
0.6
0.4
0.2
0
20000
40000
60000
80000
100000

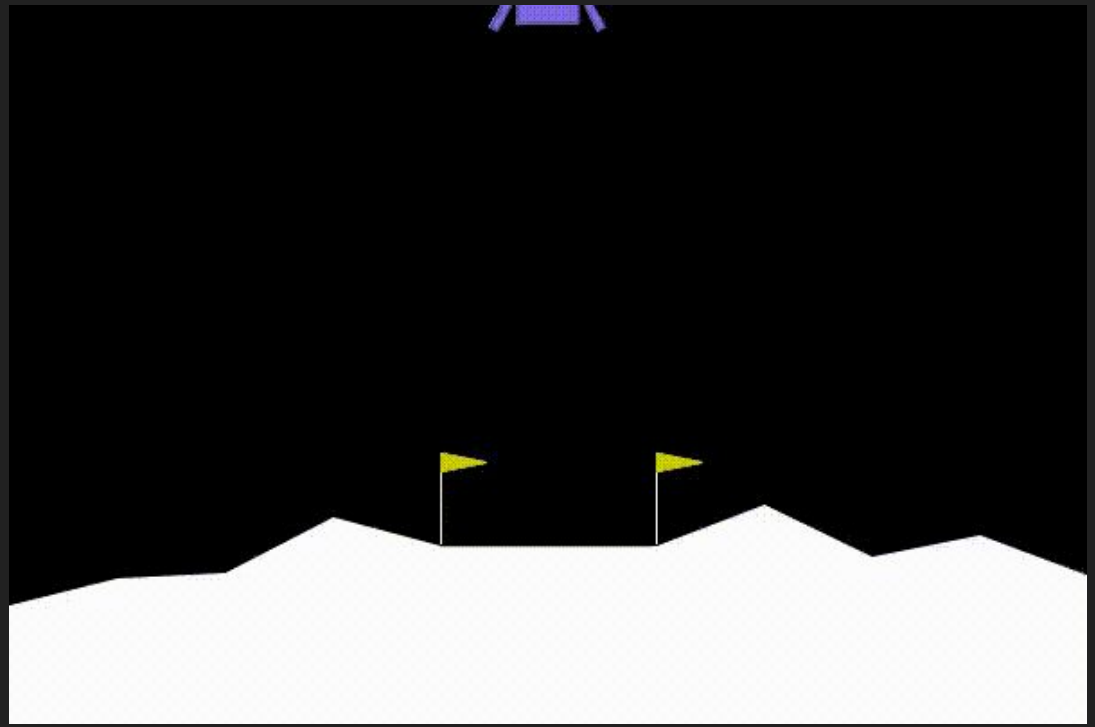
Parametr n


Rewards for DQN y=0.1 and d=0.9997
n=50
n=100
n=150
Reward
Epoch
−25
−50
−75
−100
−125
−150
−175
−200
0
20000
40000
60000
80000
100000

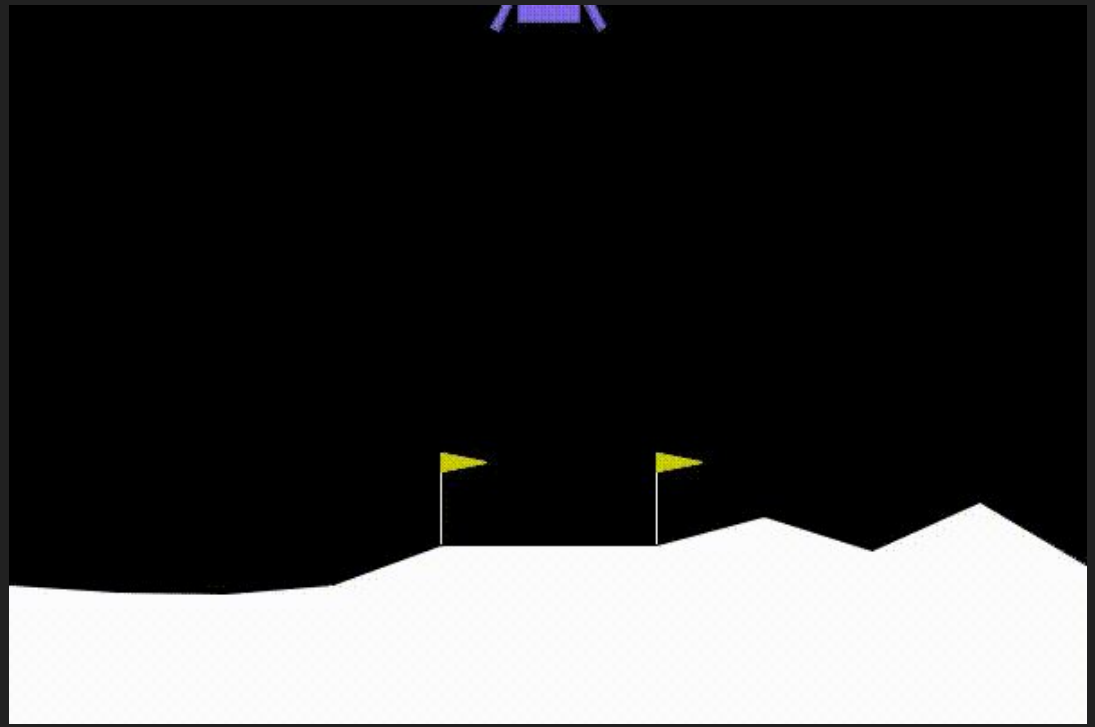
Q-learning vs deep Q-learning



Rezultat



Rezultat bez kary za paliwo



Czasy nauk

- Q-learning: 1h / 100000 epok
- Deep Q-learning: 5h / 100000 epok
- Deep Q-learning z dużym modelem: 10h / 100000 epok lub więcej

Dużym problemem w deep Q-learning jest ciągłe wywoływanie funkcji forward co każdą klatkę gry, nawet do 500 razy na epokę.

Przydatne linki:

- https://gymnasium.farama.org/environments/box2d/lunar_lander/
- https://github.com/MrKrisuuu/ReinforcementLearning