Jan Jędrzejewski
28.01.2022r

WSI - zadanie 6

Uczenie się ze wzmocnieniem

Algorytm Q-learning

Treść zadania:
Zaimplementuj algorytm Q-learning, zbadaj wpływ hiper parametrów na działanie algorytmu (np. wpływ strategii eksploracji, współczynnik uczenia).

Implementacja:
W pliku *qlearning.py* zaimplementowano funkcję *qlearning* przyjmującą jako parametry:

- liczbę epizodów,
- maksymalną liczbę kroków,
- poziom eksploracji, D: (0;1]
- współczynnik uczenia D: (0;2]
- zniżkę - współczynnik istotności kary D: (0;1]

Tworzy i uczy tablicę Q funkcji dla danego środowiska. Funkcja *test_qtable*, pozwala sprawdzić wygenerowaną tablicę, a funkcja *print_frames* wyświetla na ekranie animację z działania algorytmu.

Poniżej tabela przedstawia skuteczność algorytmu w zależności od liczby epizodów które zostały przeprowadzone w procesie uczenia. Hiper Parametry dobrane zostały metodą inżynierską - max steps=500, exploration=0.8, learning rate=1, discount=0.8. Poniższe wyniki zostały otrzymane dla próby testującej składającej się z 1000 epizodów i maksymalnie 200 kroków.

Number of episodes	avg_steps	avg_reward	% of success
100	163.092	-158.997	19.5%
250	78.638	-65.072	64.6%
500	31.884	-13.005	89.9%
1 000	13.882	7.013	99.5%
5 000	13.036	7.964	100.0%
10 000	13.057	7.943	100.0%
50 000	13.061	7.939	100.0%
100 000	13.013	7.987	100.0%

Już dla 5 000 epizodów otrzymujemy bardzo dobre wyniki, dojścia do celu na poziomie 100%. W kolejnych iteracjach nie widać wielkiego postępu, jeśli chodzi o średnią liczbę kroków lub średnie nagrody za przejście. Dlatego do badania hiper parametrów będę wykorzystywał liczbę epizodów=5 000.

Wpływ hiper parametrów:

1) Poziom eksploracji

Number of episodes;	avg_steps;	avg_reward;	percentage of success;	penalties
0.1	13.081	7.919	100.0%	0
0.3	13.036	7.964	100.0%	0
0.5	13.177	7.823	100.0%	0
0.7	13.037	7.963	100.0%	0
0.8	13.105	7.895	100.0%	0
0.9	13.058	7.942	100.0%	0
1	13.359	7.641	100.0%	0

Poziom eksploracji wypada podobnie jeśli chodzi o skuteczność, ale wraz ze wzrostem wartości tego parametru maleje czas uczenia tablicy Q. Dlatego najlepiej używać parametru powyżej 0,5. Dla exploration=1 uzyskałem najgorszy rezultat, prawdopodobnie brakuje algorytmowi eksploatacji.

2) Współczynnik uczenia

learning_rate;	avg_steps;	avg_reward;	percentage of success;	penalties
0.1	20.167	0.014	96.1%	0
0.5	13.06	7.94	100.0%	0
0.7	13.083	7.917	100.0%	0
1	12.973	8.027	100.0%	0
1.2	13.13	7.87	100.0%	0
1.6	187.168	-1193.854	6.9%	112015
2	200.0	-1027.136	0.0%	91904

Algorytm najlepsze wyniki otrzymywał dla wartości learning rate od 0,5 do 1. W tym zakresie prawie nie było różnicy. Dla wartości mniejszych brakowało douczenia funkcji Q, a dla wartości większych niż 1,3 uczenie praktycznie nie działało.

3) Zniżka

discount;	avg_steps;	avg_reward;	percentage of success;	penalties
0.1	13.133	7.867	100.0%	0
0.3	13.035	7.965	100.0%	0
0.5	13.075	7.925	100.0%	0
0.7	13.217	7.783	100.0%	0
0.8	12.957	8.043	100.0%	0
0.9	13.048	7.952	100.0%	0
1	12.95	8.05	100.0%	0

Dla tego parametru znowu otrzymujemy bardzo dobre rezultaty dla całego przekroju wartości. Nieco lepsze rezultaty otrzymałem dla większych wartości discount.

Wnioski:

Zaimplementowany algorytm Q-learning ze strategią zachłanną nie jest wrażliwy na parametry i dla niewielkiej liczby epizodów w 100% zbadał całe środowisko i wyznaczył nie mylącą się bardzo optymalną tablicę funkcji Q.