Lena Zubik

Temat: Uczenie się ze wzmocnieniem – algorytm Q-Learning

Opis algorytmu

Q-Learning to jeden z algorytmów uczenia się ze wzmocnieniem. Nie wymaga modelu środowiska i może poradzić sobie z problemami z przejściami stochastycznymi i nagrodami bez konieczności adaptacji. W wykonanej implementacji agent bazując na otrzymywanych nagrodach/karach za wykonaną akcję nauczy się wyznaczać najkrótszą ścieżkę w labiryncie.

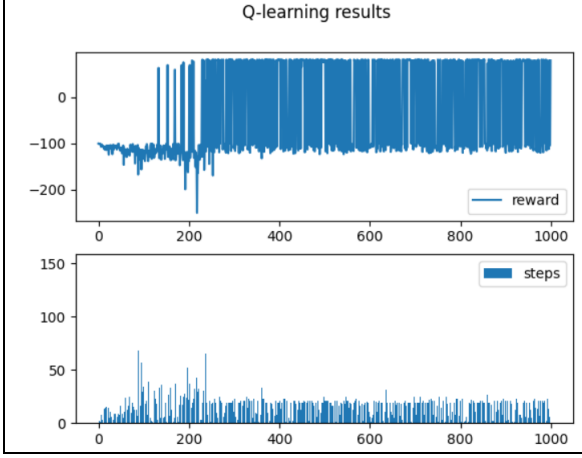
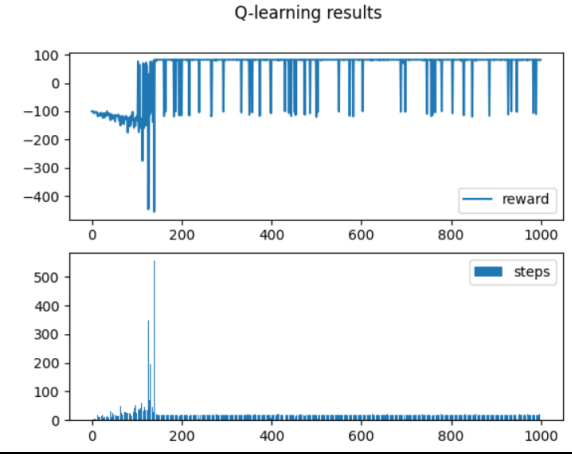
Agent na podstawie otrzymywanych nagród (-1 za każdy krok w miejsce bez przeszkody, -100 za wejście w przeszkodę, 100 za dotarcie do celu) modyfikuje konkretną wartość w tabeli q. Tabela q zbudowana jest w taki sposób , aby pokrywać kombinację dowolnych przestrzeni (w tym przypadku są to konkretne pola na planszy dwuwymiarowej) i akcji (w tym przypadku są 4 akcje: ruch w górę, ruch w prawo, ruch w dół i ruch w lewo). Zachowanie algorytmu można modyfikować, manipulując parametrami: epsilon (jak często wybierze ruch zupełnie losowy – większa eksploracja), discount rate (jak bardzo przyszłościowo agent ma wybierać ruchy), learning rate (jak bardzo za każdym razem ma zostać modyfikowana wartość w taeli q). Na podstawie tych wartości konkretna wartość w tabeli q modyfikowana jest zgodnie ze wzorem:

$$Q^{new}(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{old value}} + \underbrace{\alpha}_{\text{learning rate}} \cdot \overbrace{\Big(\underbrace{r_t}_{\text{reward}} + \underbrace{\gamma}_{\text{discount factor}} \cdot \underbrace{\max_a Q(s_{t+1}, a)}_{\text{estimate of optimal future value}} - \underbrace{Q(s_t, a_t)}_{\text{old value}} \Big)}^{\text{temporal difference}}$$
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{new value (temporal difference target)}}$$

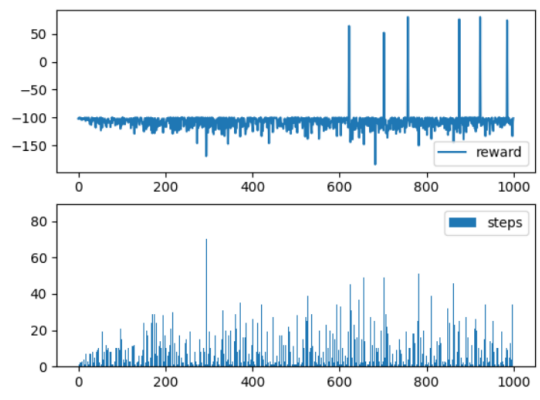
Rezultat

```
...#############
...#..S..#..#..#
#..####..#..#..#
#........#.....#
#..#..#..#..#..#
#..#..#.....#..#
#..####..#..####
#..#.F#..#.....#
#..#..########..#
#.....#.........
#############...
```

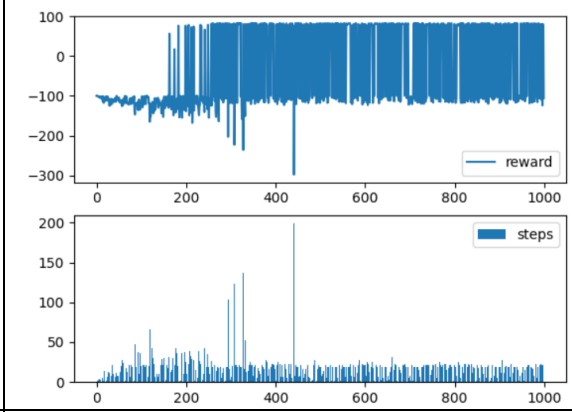
Przykładowy labirynt do analizy rezultatów

eps=0.9 discount_rate=0.9 learning_rate=0.9	eps=0.98 discount_rate=0.9 learning_rate=0.9
	

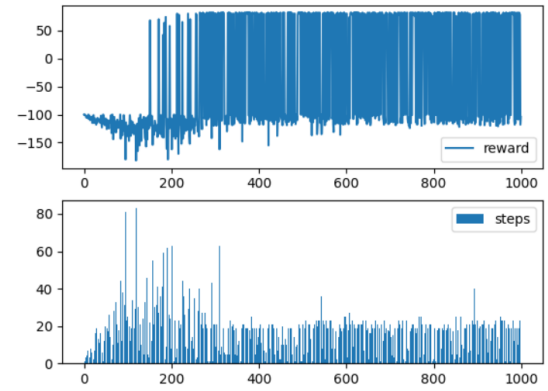
eps=0.65 discount_rate=0.9 learning_rate=0.9



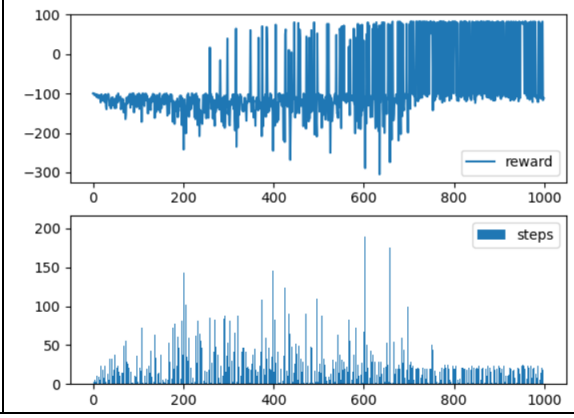
eps=0.9 discount_rate=0.99 learning_rate=0.9



eps=0.9 discount_rate=0.1 learning_rate=0.9



eps=0.9 discount_rate=0.9 learning_rate=0.1



Przy wartości współczynnika epsilon=0.98 została ograniczona liczba losowych ruchów wykonywanych przez agenta. Model po około 150 epizodzie osiąga zadaną skuteczność. Zmniejszenie tego współczynnika do wartości 0.65 spowodowało zwiększenie losowego zachowania się agenta. W efekcie wytrenowany model znalazł ścieżkę do punktu docelowego tylko w 6 epizodach na 1000 (nie była ona optymalna o czym świadczy wynik daleki od wartości 100). Ustawienie wartości discount_rate=0.99 spowodowało, że model po 200-nym epizodzie często znajdował bardzo krótką ścieżkę do celu. Obniżenie wartości tego współczynnika poskutkowało wydłużeniem odnalezionej trasy. Znaczne obniżenie współczynnika learning_rate do wartości 0.1 powoduje, że model potrzebuje więcej epizodów do nauki, w tym przypadku było to 640 epizodów. Jest to wartość znacznie większa od początkowych 150.

Wnioski

Budowa dobrego modelu opiera się na kompromisie pomiędzy tym, jak bardzo agent ma zbadać środowisko (eksploracja), jak bardzo jego ruchy mają uwzględniać strategię najszybszego dotarcia do celu oraz jak bardzo rezultaty za podjęte akcje mają wpływać na zachowanie modelu. Moim zdaniem dla zbudowanego środowiska zadawalające rezultaty zostały otrzymane dla wartości współczynników: epsilon=0.9, dicount_rate=0.9, learning_rate=0.9.