Algorytm Q-learning

Marcin Połosak

Styczeń 2024

1 Opis algorytmu

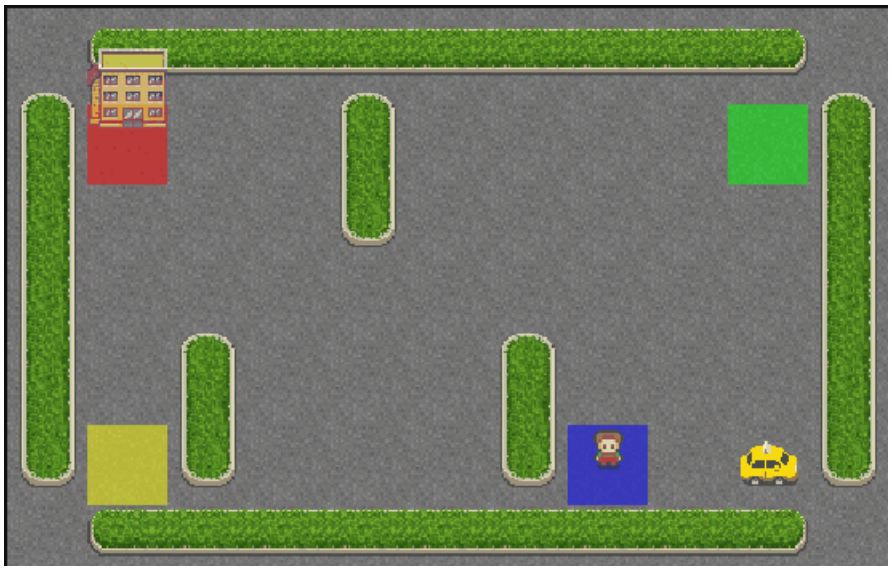
Algorytm Q-learning jest podstawowym algorytmem uczenia ze wzmocnieniem, który opiera się na poprawianiu zdolności rozwiązywania zadania z kolejnymi iteracjami nauki.

1.1 Agent

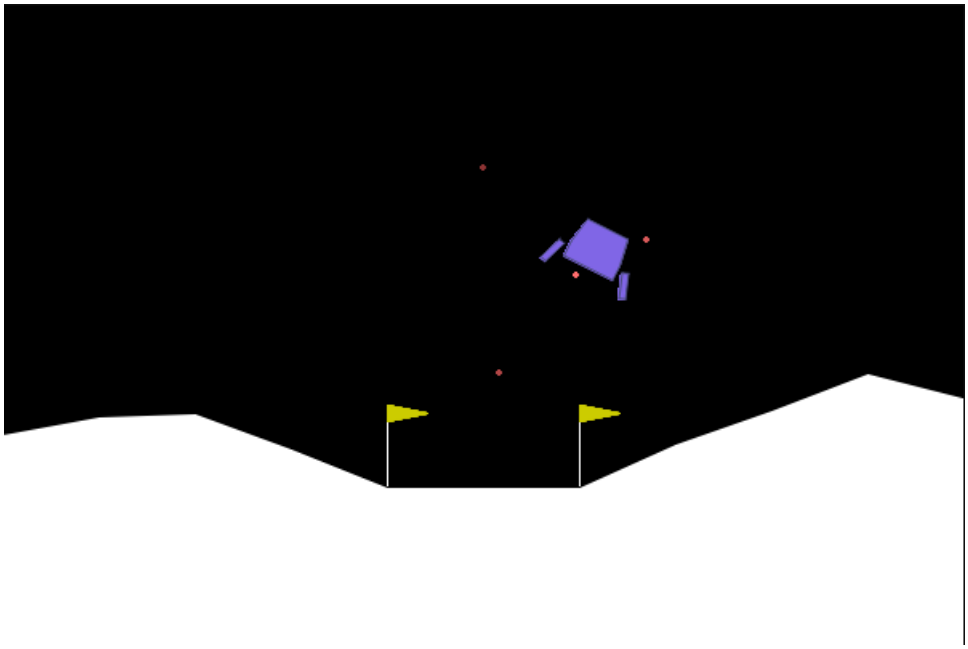
Agent to obiekt, który podlega nauczaniu. W celu poprawnego uczenia, musi posiadać zdefiniowaną przestrzeń akcji możliwych do wykonania. W ramach każdej iteracji, algorytm wybiera jeden stan do wykonania.

1.2 Stan

Stan to numeryczne określenie położenia, w którym znajduje się agent. W przypadku zadania taksówki jest on określony za pomocą pojedyńczej liczby.



W bardziej złożonych zadaniach czasami nie das się ograniczyć liczby stanów do skończonej, małej liczby. W przypadku próby uczenia lądownika kosmicznego konieczna była dyskredyzacja wektora stanów, który przekazywał informacje o takich wartościach jak położenie oraz przędkość spadku i obrotu agenta.



1.3 Uczenie

Uczenie agenta polega na wprowadzeniu oceny akcji, które może podjąć na podstawie możliwych konsekwencji. Wartości ocen zapisywane są w tabelach, które aktualizuje się za pomocą wzoru:

$$Q(state_t, action_t) = (1 - \alpha) \cdot Q(state_t, action_t) + \alpha \cdot [(R_{t+1} + \gamma \cdot \max_a(S_{t+1}, a)]$$

gdzie:

α - współczynnik uczenia, wpływ nowej wartości na tą już wpisaną,
R_{t+1} - nagroda uzyskana w stanie następnym po podjęciu danej akcji;
γ - współczynnik wpływu przyszłych wartości na ocenę obecnego stanu.

Zakłada się, że dla skończonej liczby możliwych do uzyskania stanów powinno się przyjmować wartość współczynnika uczenia bliską 1.

1.4 Strategie eksporacji

Można zauważyć, że w podstawowej wersji algorytmu będzie miał on tendencje do podążania jedną trasą, która została już sprawdzona. Można to korygować próbując zmieniać wartości tablicy podczas inicjalizacji nowych stanów, ale może to spowodować znaczące spowolnienie algorytmu, który wówczas zacznie na przykład sprawdzać wszystkie możliwe stany.

Alternatywą jest wprowadzenie elementu losowości, który sprawi, że algorytm będzie miał matematyczne szanse na wybranie niepoprawnej drogi podczas uczenia, a więc do przeszukiwania różnych dróg. W swojej pracy zbadałem trzy warianty takich strategii:

1.4.1 Metoda zachłanna

Metoda ta opiera się na założeniu, że dla ustalonej wartości ϵ algorytm będzie miał (1- ϵ) szans, że podejmie decyzje na podstawie tablicy oraz ϵ szans, że będzie to wybór losowy.

1.4.2 Metoda liczenia dla wariantu zachłannego

Modyfikacja ta, zakłada że co ustaloną liczbę iteracji wartość epsilonu będzie obniżana o przyjętą wartość. Spowoduje to zmniejszenie losowości treningu algorytmu w późniejszych etapach jego nauki. Zdobytą wiedzę będzie on już mógł wykorzystać do poprawnej aktualizacji tablicy.

1.4.3 Metoda Boltzmanna

Metoda ta, określa prawdopodobieństwo wykonania akcji na podstawie jej oceny w tablicy. Dzięki temu prawdopodobieństwo wykonania akcji lepszej zawsze będzie większe od pozostałych, ale algorytm dalej będzie czasami wybierał rozwiązania najgorsze w celu szukania innych dróg. Sposób określania prawdopodobieństwa sprowadza się do wzoru:

$$\pi(x,a) = \frac{e^{\frac{Q(x,a)}{T}}}{\sum_b \frac{Q(x,b)}{T}}$$

Wartość parametru T jest dobierana przez użytkownika.

2 Planowane eksperymenty

W ramach eksperymentów zbadam poprawność działania algorytmu badając liczbę iteracji potrzebnych do uzyskania wyniku oraz zmianę łącznej oceny podejścia w kolejnych próbach podczas uczenia.

Dodatkowo przebadam wpływ zmiany współczynnika uczenia oraz czynnika osłabiającego na końcowy rezultat.

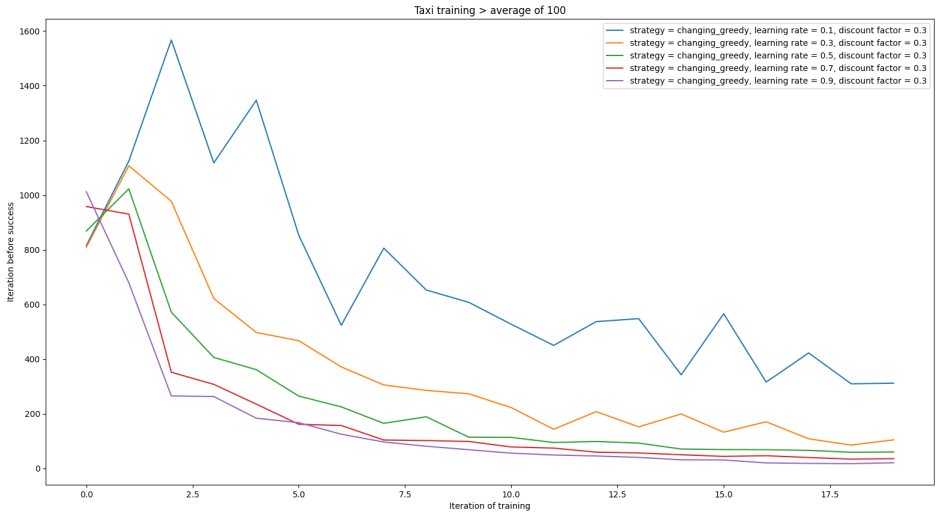
Podjęte próby wykonam dla każdej z opisanych metod eksploracji. Postaram się zaprezentować również wpływ wartości ich współczynników na wynik.

Celowo ograniczę maksymalną ilość iteracji, żeby uzyskany wynik nie był idealny, ale pokazywał, które z opcji potrafią wyznaczyć poprawny wynik szybciej. W celu dokładniejszej obserwacji charakterystyki algorytmu wykresy są wynikiem uśrednionego wyniku z 50 i 100 podejść.

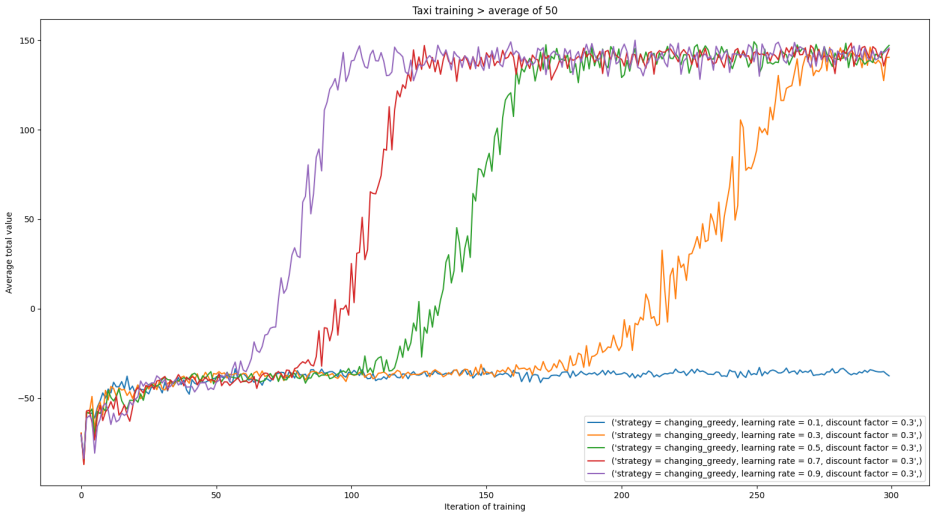
3 Rezultaty

3.1 Wpływ współczynnika uczenia

3.1.1 Liczba potrzebnych iteracji



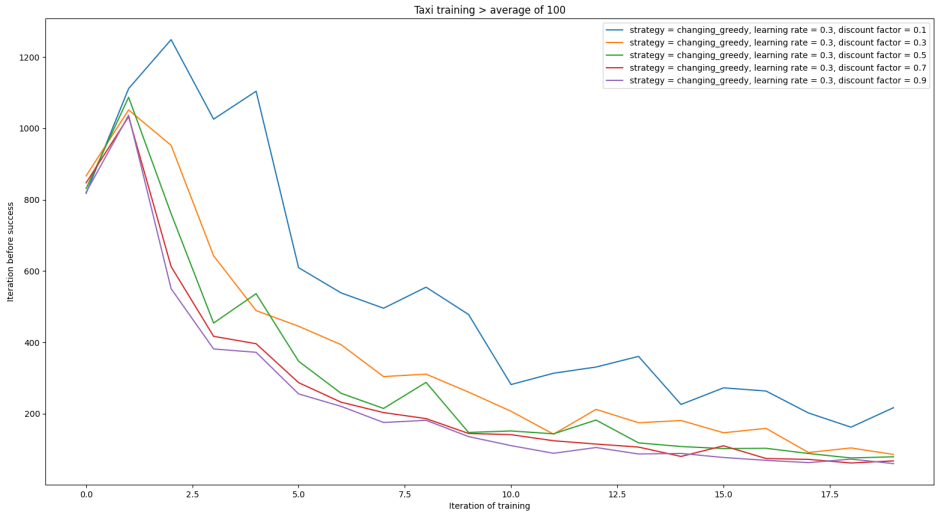
3.1.2 Średnia wartość oceny



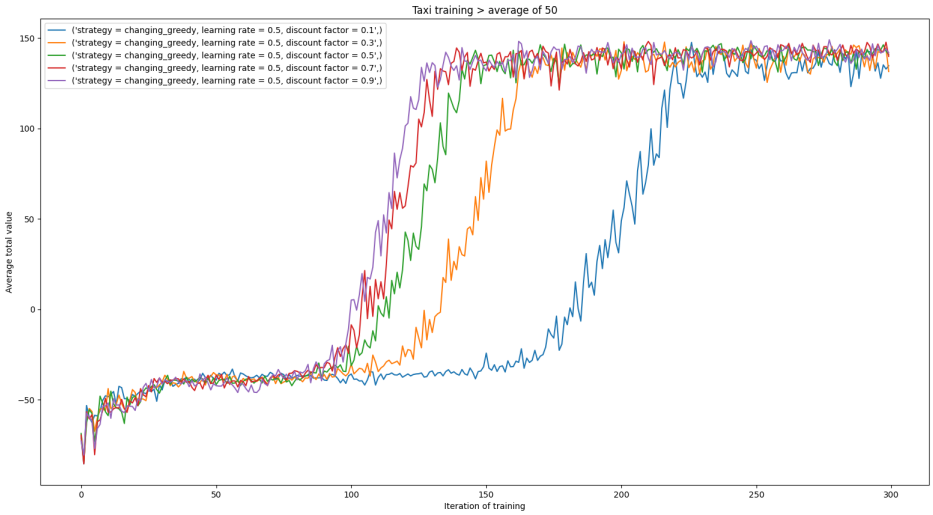
Można zauważyć, że im większa wartość parametru uczenia tym szbciej udaje się uzyskać najlepszy wynik. Dla badanego zadania taksówki potwierdza to założenie mówiące, że dla ograniczonej ilości stanów większy współczynnik będzie najlepszy.

3.2 Wpływ czynnika osłabiającego

3.2.1 Liczba potrzebnych iteracji



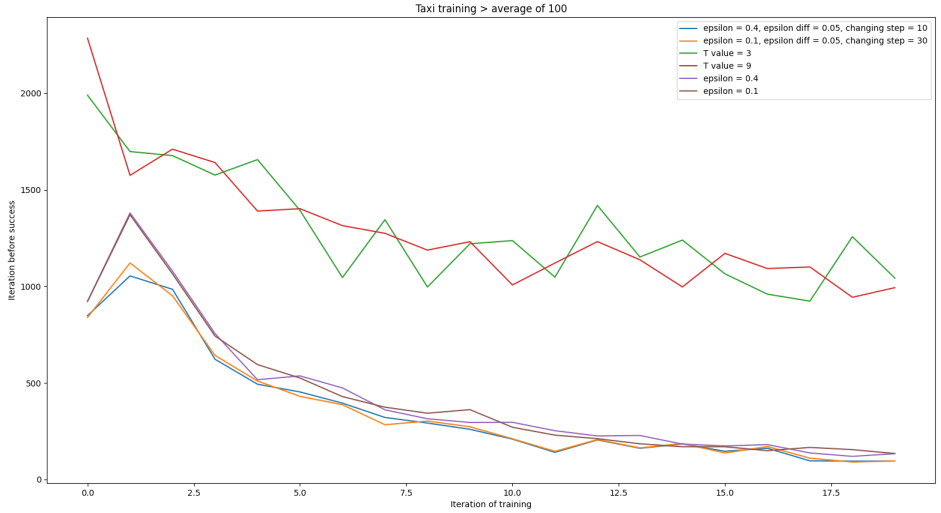
3.2.2 Średnia wartość oceny



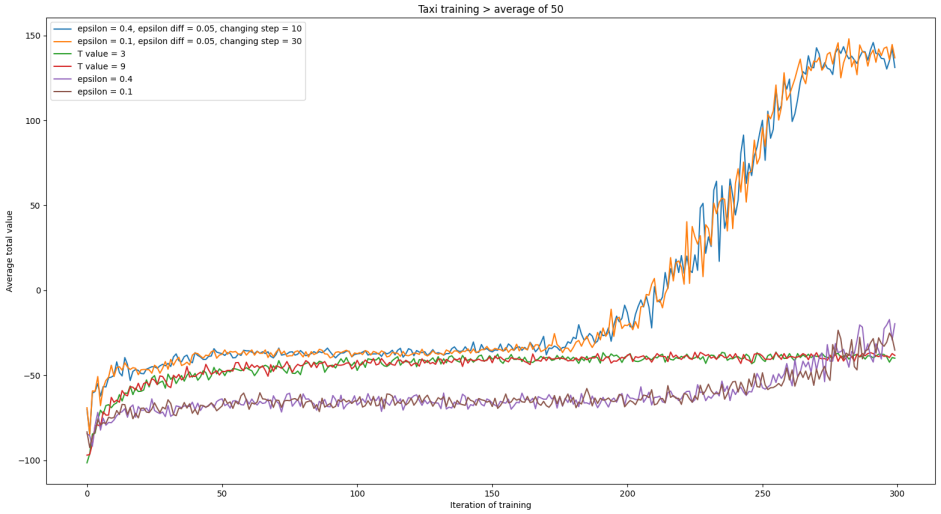
Również większy czynnik osłabiający wydaje się mieć lepsze rezultaty. Wydaje sie być to efektem prostoty oceny stanu, która gratyfikuje agenta tylko w przypadku osiągnięcia celu zadania. W przypadku, gdy po drodze występywowałyby mniejsze nagrody, współczynnik ten też powinien być dobierany mniejszy.

3.3 Porównanie różnych wariantów eksploracji

3.3.1 Liczba potrzebnych iteracji



3.3.2 Średnia wartość oceny



Ciężko wyłonić najlepszą strategie eksploracji dla tak określonego zadania. Wydaje się, że dla tak przyjętych parametrów najlepsze efekty uzyskała strategia stopniowego zmniejszania losowości algorytmu. Nie znaczy to jednak, że będzie to również najlepsza opcja dla bardziej złożonych zadań.

4 Wnioski

Dla przebadanego zadania taksówki, skończona liczba około 500 stanów spowodowała względnie szybkie wyznaczenie najszybszej trasy niezależnie od przyjętej strategii oraz parametrów.

Dla bardziej złożonych zadań, istotone jest dokładne wyznaczenie hiperparametrów w celu zagwarantowania zbiegania algorytmu do wartości optymalnej.

Udało się zaimplementować algorytm, który jest w stanie znajdywać rozwiązanie zadanie po stosunkowo nie dużej liczbie iteracji treningowych.

Początkowa losowość uczenia jest uzasadniona w celu znalezienia wszelkich możliwych dróg spełniających zadanie.

Dla bardziej złożonych zadań dobrą praktyką jest wykorzystanie dyskretyzacji stanów lub na przykład sieci neuronowych w celu zagwarantowania przejrzenia większości stanów z przestrzeni, która stanowi potencjalną drogę do rozwiązania.