Q-Learning

Kacper Kania
Wydział Informatyki i Zarządzania
Politechnika Wrocławska
220873@student.pwr.edu.pl

Michał Kosturek
Wydział Informatyki i Zarządzania
Politechnika Wrocławska
220885@student.pwr.edu.pl

Streszczenie

Reinforcement learning to jeden z podstawowych problemów w dziedzinie uczenia maszynowego. W skład algorytmów, które starają się rozwiązać ten problem są Q-Learning oraz Deep Q-Learning. W trakcie tego laboratorium opisano obie metody. Zbadano je również pod względem zdefiniowanych metryk na grze „kółko i krzyżyk". Następnie obie metody zestawiono ze sobą i porównano liczbę zwycięstw, przegranych oraz remisów dla każdego z nich. Kolejno pokazano działanie algorytmów zestawieniu z graczem losowym. Otrzymane wyniki pokazują, że obie metody są znacząco lepsze od gracza losowego, natomiast Deep Q-Learning jest lepszą metodą od podstawowego Q-Learningu ze względu na jego stopień generalizacji decyzji dla różnych stanów gry, których mógł podczas nauki nie odwiedzać.

1 Wstęp

Reinforcement learning jest jednym z problemów zdefiniowanych w dziedzinie sztucznej inteligencji. Polega na nauczeniu agenta polityki działania w środowisku nieznanym dla niego. Sposób, w jaki agent poznaje świat, odbywa się poprzez system nagród i kar. Kiedy agent wykona akcję, która przybliży go do jego celu, otrzymuje nagrodę. W przypadku, gdy akcja nie powinna być wykonana, dostaję karę. Przy wykorzystaniu takiej metody po pewnym czasie kształtuje swój obraz na temat świata, w którym się znajduje. Dzięki temu wie, jak może się w nim poruszać oraz jakie akcje są niedozwolone.

W kontekście uczenia ze wzmocnieniem istnieje wiele zbadanych metod pozwalających na tworzenie obrazu świata i polityki działania w tym świecie. Jednym z ważniejszych algorytmów jest Q-Learning. W związku z dużym rozwojem Q-Learningu, powszechnie wykorzystywaną metodą jest także Deep Q-Learning (DQL), który zamiast zapamiętywania tablicy stanów środowiska wykorzystuje sieci neuronowe do reprezentacji wiedzy o środowisku i operuje bezpośrednio na zakodowanym stanie gry. Obie metody wykorzystano w trakcie przeprowadzania tego laboratorium.

2 Wykorzystane metody

W tym rozdziale przedstawiono opis algorytmów Q-Learning oraz Deep Q-Learning.

2.1 Q-Learning

$$Q^{new}(s_t, a_t) \leftarrow (1-\alpha) \cdot Q^{old}(s_t, a_t) + \alpha\Big(r_t + \gamma \cdot Q(s_{t+1}, a)\Big)$$

gdzie:

- α - współczynnik uczenia
- γ - współczynnik dyskontu (opóźnienia nagrody)
- Q - tablica (stan, akcja) $\rightarrow$ wartość funkcji oceny
- s_t - obecny stan gry
- a_t - akcja podjęta w obecnym stanie gry
- r_t - nagroda za podjęcie akcji a_t w stanie s_t
- $s_{(t+1)}$ - stan po podjęciu akcji a_t w stanie s_t

2.2 Deep Q-Learning

1. Inicjalizacja funkcji Q - sieci neuronowej z losowymi wagami
2. Dla epizodów od 1 do N:
   (a) Rozpocznij nową grę
   (b) Do zakończenia gry, dla każdej tury t:
   - Z prawdopodobieństwem ϵ_{epizod} wybierz losową akcję a_t, w przeciwnym przypadku wybierz $a_t = \arg\max_a Q(s_t, a)$
   - Wyznacz nagrodę r_t dla stanu s_{t+1} po wykonaniu akcji a_t w stanie s_t
   - Stwórz kolekcję stanów planszy i ruchów jednoznacznych z s_t (rotacje, transpozycje, odbicia symetryczne)
   - Oczekiwana wartość funkcji Q to: $y_t = \begin{cases} r_t & \text{jeśli } s_{t+1} \text{ to stan końcowy} \\ r_t + \gamma \cdot \max_a Q(s_{t+1}, a) & \text{jeśli } s_{t+1} \text{ w p.p.} \end{cases}$
   - Wykonaj aktualizację sieci realizującej funkcję Q na wygenerowanym *batchu* względem błędu średniokwadratowego $\big(y - Q(s_t, a_t)\big)^2$.

2.3 Strategia epsilonowa

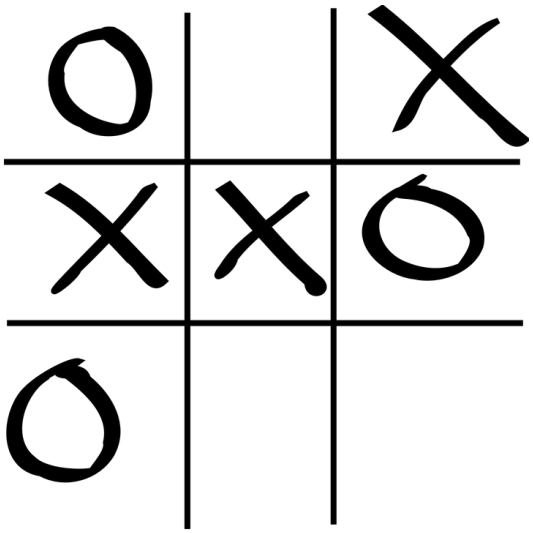
Zarówno przy uczeniu Q-Learning jak i Deep Q-Learning zastosowano wygaszaną strategię epsilonową. Oznacza to, że w każdym kolejnym epizodzie nauki prawdopodobieństwo wykonania ruchu losowego (a więc eksploracji przestrzeni stanów gry) jest zmniejszane, na rzecz wykonywania ruchów zgodnych z wyuczaną strategią (eksploatacji przestrzeni stanów).

3 Zdefiniowany problem

W tym rozdziale przedstawiono zdefiniowany problem, dla którego badano metody Q-Learningu oraz Deep Q-Learningu.

3.1 Opis

W trakcie badania obu metod wykorzystano popularną grę „kółko i krzyżyk". Polega ona na zapełnieniu planszy symbolami. Wygrywa ten gracz, który swoje symbole ustawi w kombinacji 3 identycznych symboli po kolei w pionie, poziomie lub po skosie. Przykładową, niezakończoną rozgrywkę przedstawiono na rysunku 1. Mimo prostoty gry, jest ona podstawowym przykładem, dla którego można badać różne metody uczenia maszynowego w kontekście prowadzenia rozgrywki.



Rysunek 1: Przykładowa rozgrywka gry „kółko i krzyżyk”

3.2 Reguły gry

1. Generowana jest pusta, kwadratowa plansza.
2. Każdy z graczy przyjmuje losowo jeden z symboli: **X** lub **O**.
3. Losowo jest generowany symbol rozpoczynający.
4. Gra przebiega turowo. Zmiana tury następuje po wykonaniu ruchu przez aktualnie ruszającego się gracza.
5. Gracz może położyć swój symbol na jednym z niezajętych pól.
6. Dany gracz wygrywa, jeżeli po położeniu swojego symbolu występuje kombinacja horyzontalna, wertykalna lub skośna jego symboli w określonej liczbie (najczęściej 3 dla planszy 3×3).
7. Może dojść do remisu, jeżeli żaden z graczy nie jest w stanie ułożyć kombinacji, a nie ma już wolnych pól na planszy.

4 Badania

W tym rozdziale przedstawiono przeprowadzone badania oraz otrzymane wyniki dla algorytmów Q-Learning oraz Deep Q-Learning. Opisano także metodologię przeprowadzanych badań.

4.1 Metryki

- Procentowa liczba remisów w oknie przesuwnym zawierającym 100 gier
- Średnia wartość nagrody w oknie przesuwnym zawierającym 100 gier
- Strata MSE dla modelu w Deep Q-Learning
- Liczba wygranych, remisów oraz przegranych w zestawionych dwóch różnych graczy

4.2 Metodologia

W badaniach I oraz II zostały zbadane algorytmy Q-Learning w formie podstawowej implementacji oraz Deep Q-Learning. Dla obu metod zbadano średnią wartość nagrody w obrębie okna przesuwnego zawierającego 100 rozegranych epizodów w trakcie uczenia. Przy wykorzystaniu tego okna zmierzono również, jak zmieniał się procentowy udział liczby wygranych, przegranych oraz remisów w liczbie wszystkich epizodów. Dla samego Q-Learningu sprawdzono również stopień wypełnienia tablicy zawierającej pary $(stan\ gry,\ akcja)$ przy znajomości liczby wszystkich możliwych stanów w grze kółko krzyżyk dla planszy 3×3 wynoszącej 5812.

W badaniu IIIyuczonych graczy Q-Learning oraz Deep Q-Learning sprawdzono pod względem skuteczności prowadzenia rozgrywki, to jest który z graczy częściej wygrywa. Do gracza Q działającego przy pomocy tradycyjnego Q-Learningu przypisano symbol $\times$, a do gracza $DeepQ$ używającego Deep Q-Learning - symbol $\circ$. Po tym zamieniono symbole obu graczy i rozegrano kolejne 5 000 gier. Otrzymane wyniki uśredniono dla poszczególnych graczy oraz przedstawiono je w postaci tabeli oraz wykresu.

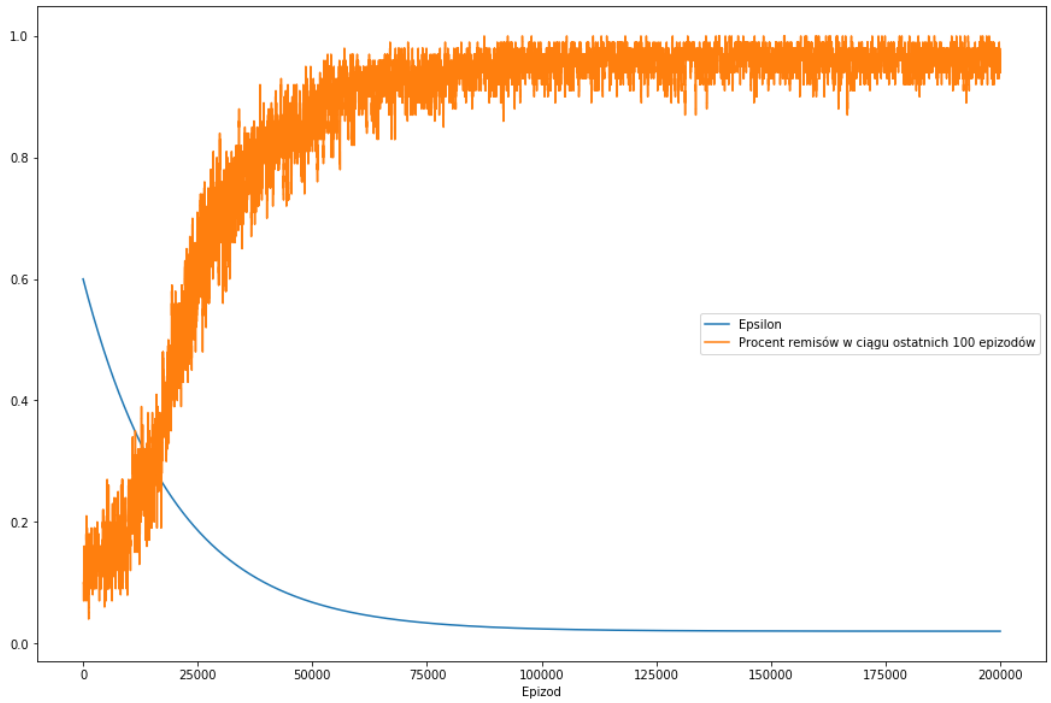
Dla badania IVestawiono wykresy procentu udziału remisów trakcie uczenia obu algorytmów.

W badaniu Vorównano skuteczność obu graczy z graczem losowym. Przydział symboli i liczbę rozegranych gier wykonano w ten sam sposób, jak przy porównaniu Q-Learning i Deep Q-Learning.

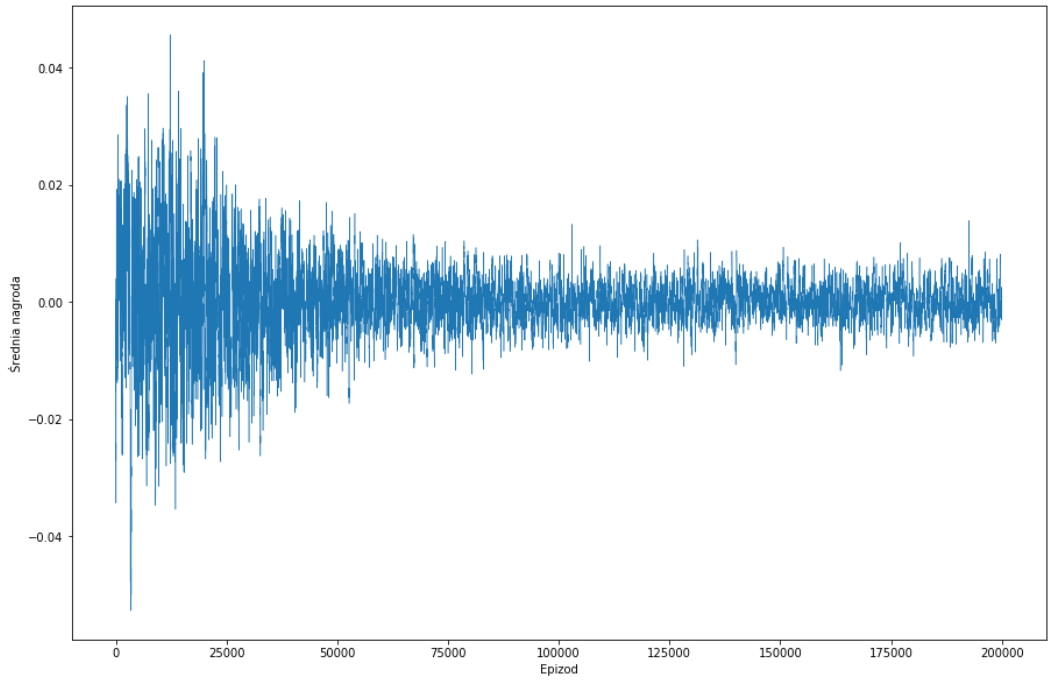
4.3 Parametry gry

1. Wielkość planszy: 3×3
2. Wartości nagród:
   - Wygrana: 1
   - Przegrana: -1
   - Remis: 0
3. Wartości ϵ:
   - Początkowa ϵ_0: 0.8 (QL), 0.6 (DQL)
   - Końcowa ϵ: 0.02 (QL), 0.1 (DQL)
   - Zbieżność do wartości końcowej: 20000 (QL) oraz 5000 (DQL) epizodów
4. Dyskont nagrody: 0.9
5. Współczynnik uczenia α Q-Learning: 0.3
6. Parametry Deep Q-Learningu:
   - Współczynnik uczenia α: 0.001
   - Architektura sieci:
     (a) Warstwa *fully connected* - 128 neuronów ukrytych
     (b) Aktywacja *ReLU*
     (c) Warstwa *fully connected* - 9 neuronów wyjściowych
     (d) Strata: błąd średniokwadratowy

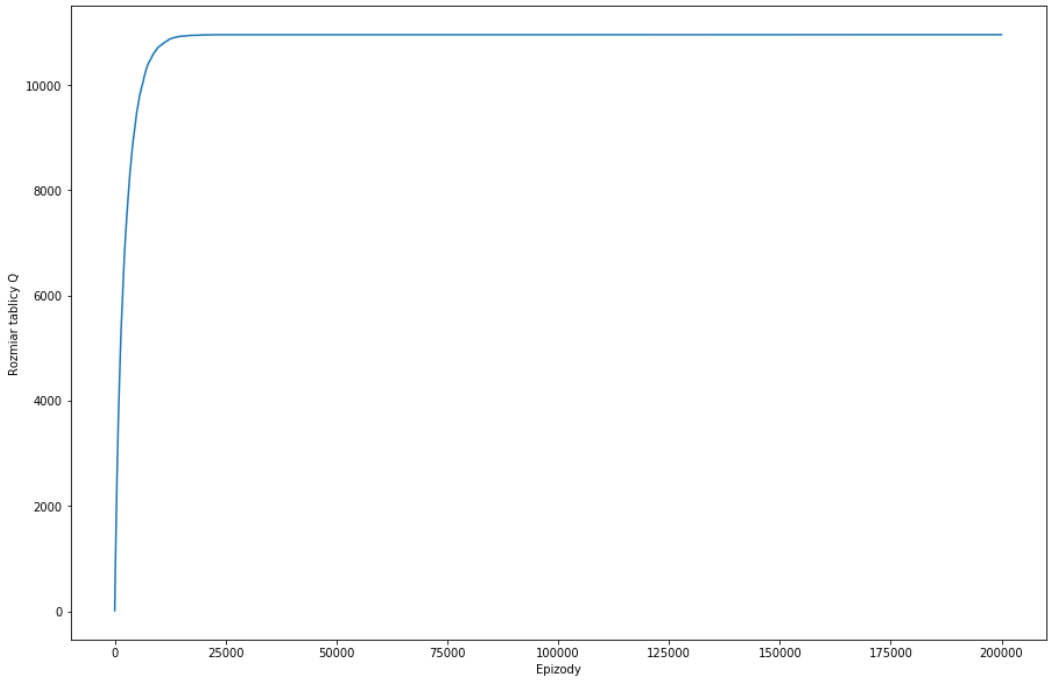
4.4 Badanie I analiza Q-Learningu



Rysunek 2: Wykres zależności wartości ϵ oraz średniego procentu remisów w oknie 100 gier od liczby epizodów

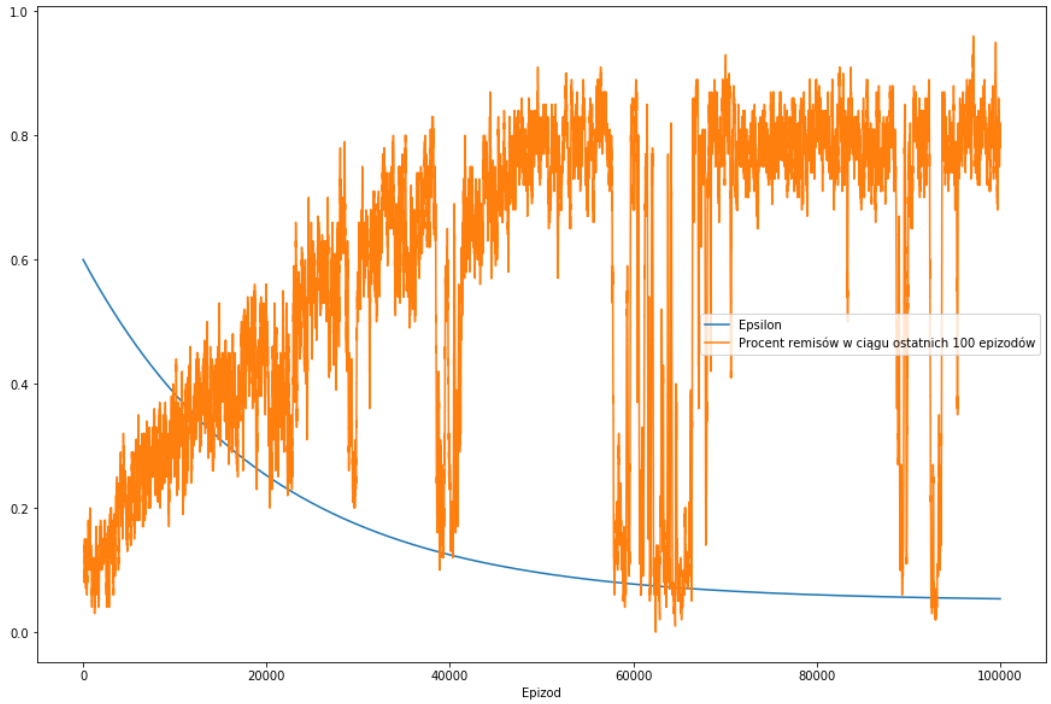


Rysunek 3: Wykres zależności średniej nagrody w oknie 100 gier od liczby epizodów

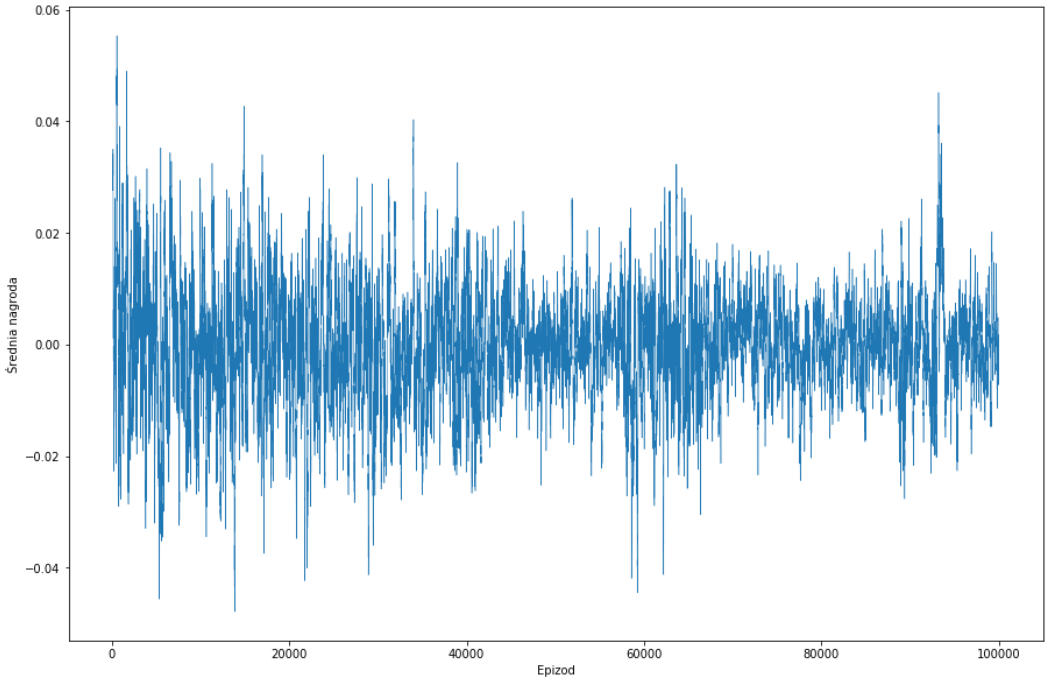


Rysunek 4: Wykres zależności liczby eksplorowanych stanów a liczbą epizodów

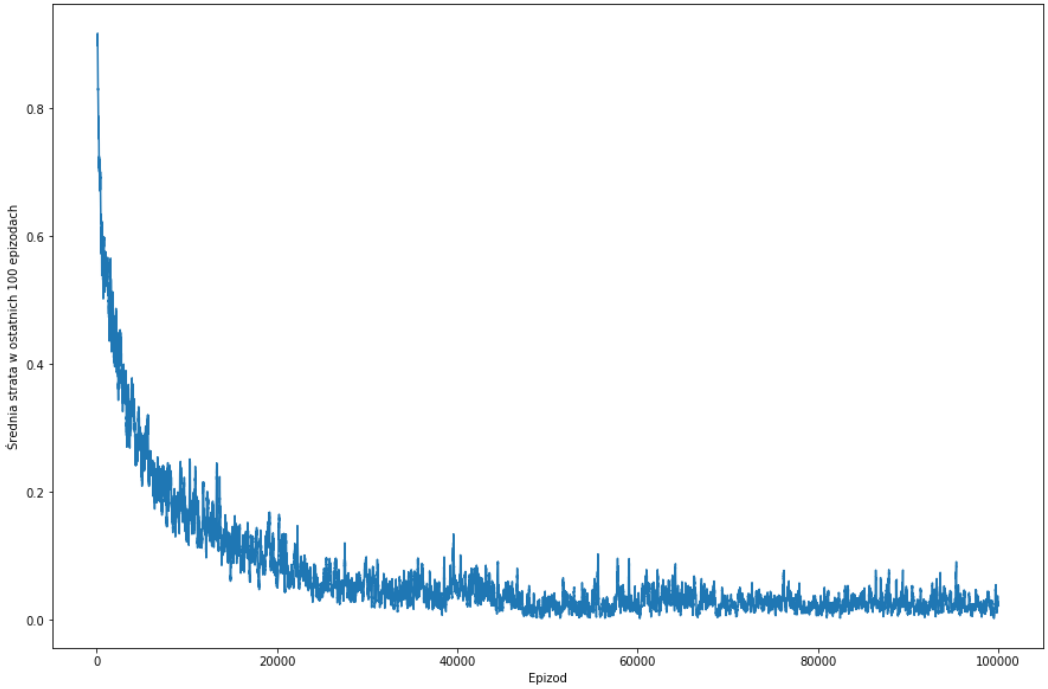
4.5 Badanie II analiza Deep Q-Learningu



Rysunek 5: Wykres zależności wartości ϵ oraz średniego procentu remisów w oknie 100 gier od liczby epizodów



Rysunek 6: Wykres zależności średniej nagrody w oknie 100 gier od liczby epizodów



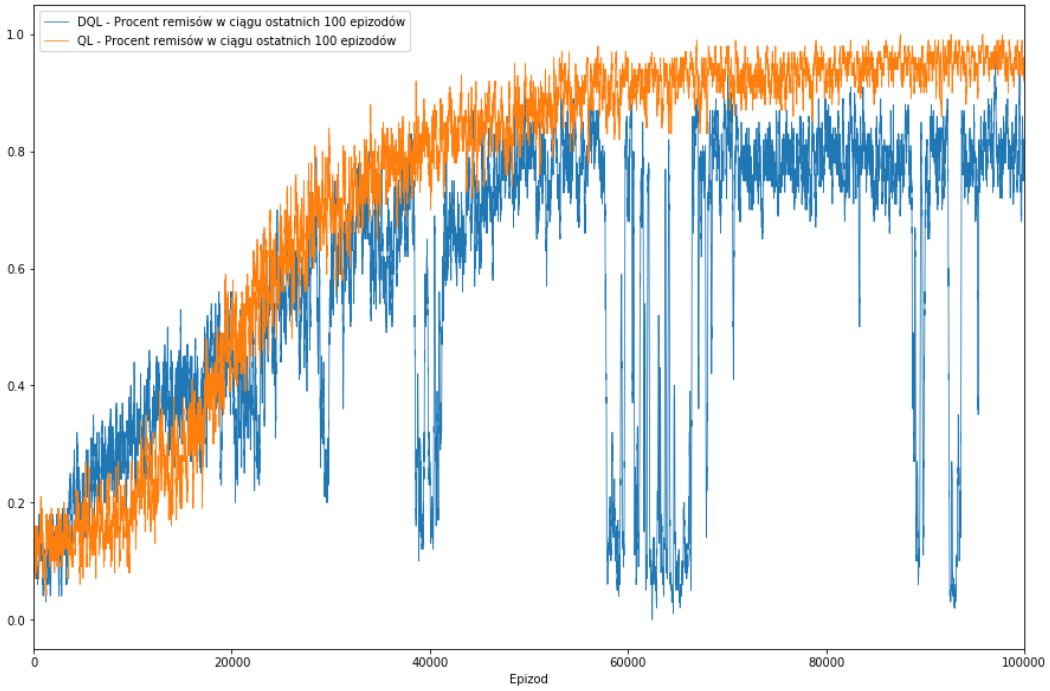
Rysunek 7: Wykres zależności straty MSE modelu od numeru epizodu dla okna przesuwnego wielkości 100 epizodów

4.6 Badanie III porównanie metod

Tablica 1: Porównanie graczy Q-Learning oraz Deep Q-Learning

Liczba rozegranych gier	Zwycięstw QL	Remisów	Zwycięstw DQL
10 000	1286	8548	166

4.7 Badanie IV zestawienia procentowego udziału remisów w rozegranych grach



Rysunek 8: Zestawienie procentowej liczby remisów w oknie przesuwnym zawierającym 100 gier dla algorytmów Q-Learning i Deep Q-Learning

4.8 Badanie V porównanie z graczem losowym

Tablica 2: Porównanie graczy Q-Learning i Deep Q-Learning w rozgrywce z graczem losowym

Gracz	Wygranych	Remisów	Przegranych
Q-Learning	7127	2716	157
Deep Q-Learning	9144	700	156

5 Podsumowanie i wnioski

W trakcie tego laboratorium zostały zbadane dwa algorytmy: Q-Learning oraz Deep Q-Learning. Wykorzystano przy tym grę „kółko i krzyżyk”. Dla każdego z algorytmów wykonano procedurę uczenia. Przedstawiono hiperparametry modeli oraz parametry gry. Następnie nauczone modele

zestawiono ze sobą oraz z graczem losowym. Rezultaty otrzymano poprzez rozegranie 10 000 gier, po 5 000 na każdy z symboli dla graczy. Otrzymane wyniki pokazano w tabelach oraz na wykresach.

Na podstawie przedstawionych wykresów można stwierdzić, że oba algorytmy zbiegły, przy czym Q-Learning charakteryzuje się szybszą zbieżnością o mniejszej wariancji. Taki gracz potrafił bardzo często doprowadzać remisów w znaczącej części wygrywać z nim. Można też zauważyć, że Q-Learning szybko osiąga stan, w którym praktycznie wszystkie możliwe stany gry są wyeksplorowane i zapisane w tablicy.

Deep Q-Learning wykazał gorszą zbieżność, grając sam ze sobą, jednak w starciu z graczem losowym częściej wygrywał niż zwykły Q-Learning. Wynika to z faktu, że taki gracz lepiej modeluje przestrzeń możliwych stanów poprzez zastosowaną sieć neuronową. Jednak w starciu z graczem Q-Learning, DQL często przegrywał. Może to wynikać ze zbyt małej liczby epok, które mogłyby lepiej symulować sytuacje występujące w takiej grze. W dodatku można zauważyć, że DQL jest bardziej wrażliwy na dobór hiperparametrów oraz jego uczenie trwa dłużej.

Ciekawą rzeczą do zauważenia, że Deep Q-Learning szybciej uczy się doprowadzać do remisu w grze. Jednak w pewnym momencie nie jest w stanie osiągnąć lepszych wyników.