Inżynieria Wiedzy i Uczenie Maszynowe

Podsumowanie projektu

Łukasz Niemiec
Michał Zakrzewski

1) Założenia i cele projektu

W ramach projektu zrealizowany został zaproponowany temat polegający na przygotowaniu modelu, opartego o metody uczenia maszynowego, umożliwiającego sterowanie robotem w środowisku Robocode. Model przeznaczony jest do sterowania robotem w walkach 1 na 1. Głównym podejściem do implementacji modelu jest wykorzystanie uczenia ze wzmocnieniem. Początkowo chcieliśmy wykorzystać również sieć neuronową, ale tego elementu projektu nie udało się zrealizować.

2) Rozwiązanie

W związku z wykorzystaniem uczenia ze wzmocnieniem konieczny jest wybór reprezentacji stanu środowiska oraz zbioru akcji robota. Ze względu na stosunkowo powolny trening związany z niską szybkością symulacji w środowisku Robocode istotne jest ograniczenie liczby możliwych stanów oraz liczby akcji.

a) Reprezentacja stanu

Stan środowiska opisywany jest przez 6 parametrów:

- energia robota
  - 3 wartości: (0, 10), [10, 50), [50, +∞)
- energia przeciwnika
  - 3 wartości: (0, 8], (8, 40], (40, +∞)
  - wartości 8 oraz 40 = 8 * 5 wynikają z obrażeń pocisku o wartości 8
- odległość do przeciwnika
  - 3 wartości: (0, 20], (20, 100], (100, +∞)
- kąt do przeciwnika
  - 4 wartości: (-45, 45], (45, 135], (135, 225], (225, 315]
- ruch przeciwnika
  - 2 wartości: ruch(prędkość przeciwnika nie mniejsza niż 1), brak ruchu

- położenie robota na arenie
    - 9 wartości: szerokość i wysokość areny podzielona na 3 przedziały w stosunku 1:2:1(jak na rysunku)

0	1	2
3	4	5
6	7	8

Dodatkowo wprowadzony został specjalny stan, reprezentujący sytuację, w której położenie przeciwnika jest nieznane.

Razem środowisko może się więc znajdować w jednym z 3 * 3 * 3 * 4 * 2 * 9 + 1 = 1945 stanów.

b) Akcje robota

Robot w każdym stanie wykonuje jedną z 6 akcji:

- obrót i ruch o ustaloną odległość(50 jednostek) w górę / w dół / w lewo / w prawo
- strzał(z siłą 2) w miejsce, w którym widziano przeciwnika
- strzał(z siłą 2) przewidujący ruch przeciwnika

c) Sterowanie robotem

Sterowanie robotem odbywa się zgodnie z paradygmatem sense-think-act, tzn. robot cyklicznie skanuje środowisko, podejmuje decyzję o następnej akcji oraz wykonuje ją. W przypadku wykorzystania uczenia ze wzmocnieniem konieczne jest również aktualizowanie wiedzy. Szczegółowe akcje robota w poszczególnych etapach to:

- sense
    - analiza wydarzeń od poprzedniej akcji(patrz punkt d))
    - obrót radaru o 90 stopni
- think
    - aktualizacja wiedzy na podstawie zebranych wydarzeń
    - zapisanie informacji o przeciwniku(o ile został wykryty)
    - wybór następnej akcji
- act
    - wykonanie wybranej akcji

d) Zbieranie informacji o wydarzeniach

Środowisko robocode informuje roboty o wielu wydarzeniach mających miejsce na polu walki. W projekcie wykorzystaliśmy następujące wydarzenia wraz z następującymi nagrodami:

- wykryto przeciwnika
    - zapisujemy położenie przeciwnika
    - brak nagrody(skanowanie nie jest akcją “wybieraną” przez agenta, lecz wykonywane jest cały czas
- uderzenie w ścianę
    - nagroda: -5
- uderzenie przeciwnika
    - nagroda: 1
- trafienie przeciwnika strzałem
    - nagroda: 10
- bycie trafionym strzałem przeciwnika
    - nagroda: -10
- śmierć
    - nagroda: -100
- zniszczenie przeciwnika
    - nagroda: 75
- nietrafienie strzałem
    - nagroda: -10

W każdej iteracji sense-think-act sumujemy i otrzymane nagrody i wykorzystujemy tę sumę do aktualizacji wiedzy(patrz punkt e)).

e) Uczenie i jego parametry

Uczenie odbywa się przy użyciu standardowego podejścia Q-Learning wykorzystując sumę nagród za ostatnią akcję jako nagrodę w danym kroku.

Parametry uczenia zmieniane są wraz z procesem uczenia. Learning rate oraz exploration rate są równe i spadają w logarytmem numeru rundy:

$$\alpha = \varepsilon = \max\{0.05,\ \min\{1.00,\ 1.00\ -\ \log_{10}\frac{round}{1000}\}\}$$

Discount factor pozostaje natomiast stały przez cały czas i ma wartość $\gamma = 0.90$

3) Aspekty implementacyjne

Ze względu na charakter projektu wykorzystaliśmy technologie Java oraz Robocode. Biblioteki do uczenia opisane zostały poniżej.

a) Model oparty o sieć neuronową

Chcieliśmy wykorzystać bibliotekę Deeplearning4j, a w szczególności jej moduł RL4J, ale problemy z dokumentacją oraz ograniczony czas spowodowały, że nie udało się zrealizować tej części projektu.

b) Model oparty o uczenie ze wzmocnieniem

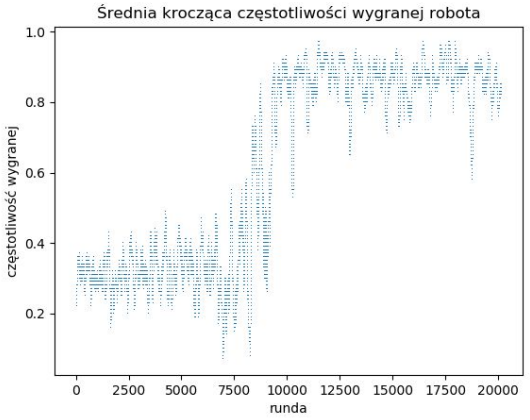
Wykorzystaliśmy bibliotekę java-reinforcement-learning (https://github.com/chen0040/java-reinforcement-learning), ze szczególnym uwzględnieniem modelu Q-Learn. Biblioteka zapewnia najbardziej podstawowe funkcjonalności konieczne do uczenia, tzn. wybór następnej akcji oraz aktualizację wiedzy. Brakuje niestety możliwości aktualizacji parametrów uczenia czy wykorzystania exploration rate przy uczeniu, zatem te zagadnienia zostały zaimplementowane w ramach realizacji projektu.

4) Wyniki

Uzyskane wyniki zależne są w znaczący sposób od wykorzystanego przeciwnika.

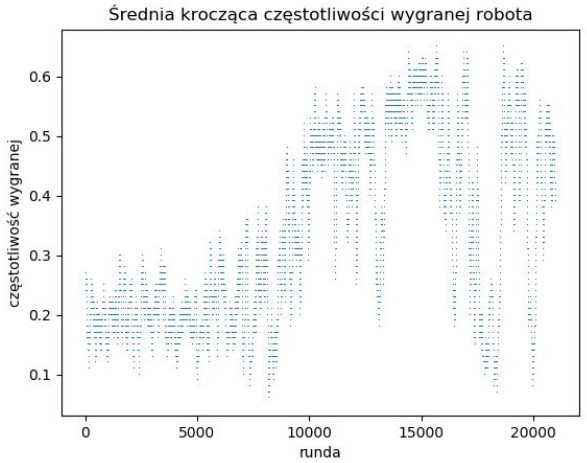
W celu łatwiejszej oceny jakości rozwiązania analizowane były wykresy średniej kroczącej(okno długości 50) częstotliwości wygranej na okresie 20000 rund. Poniżej przedstawiono kilka przykładowych wykresów uzyskanych przeciwko różnym przeciwnikom.

- wyniki przeciwko SpinBot
    - w okolicy rundy 10000 widoczny jest znaczący wzrost skuteczności robota. Robot nie radzi sobie idealnie ale jego skuteczność oscyluje w okolicy 90%, co można uznać za dobry wynik.

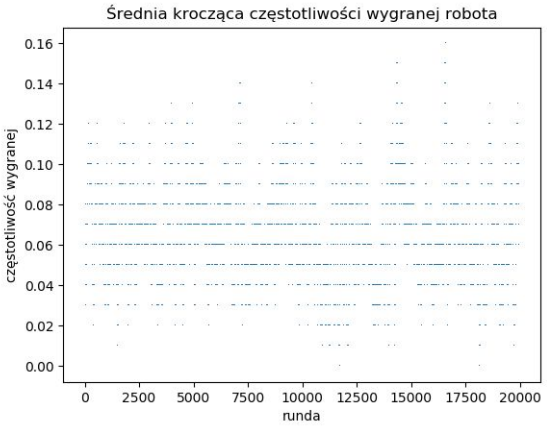


- wyniki przeciwko Target
    - Robot Target nie jest silnym przeciwnikiem, ale skuteczność uczonego robota na początku wynosiła około 80-95%, a w późniejszym etapie uzyskała wartość niemal 100%.

- wyniki przeciwko TrackFire
    - Z początkowej skuteczności na poziomie około 20%, udało się osiągnąć skuteczność na poziomie 60%, chociaż jej stabilność jest mała(znaczący, chwilowy spadek w okolicy rundy 17500).



- wyniki przeciwko Walls
    - Nie udało się osiągnąć żadnej poprawy rezultatów, możliwe że inny dobór parametrów sterowania robotem pozwoliłby uzyskać lepsze rezultaty.



Przykładowe walki robota przed i po uczeniu dostępne są pod adresami:

- przed uczeniem: https://www.youtube.com/watch?v=hfragZ8w2jI
- po uczeniu: https://www.youtube.com/watch?v=tEeN9BdIS6c

5) Podsumowanie

W ramach projektu udało się zaimplementować model, który potrafi uczyć się strategii odpowiedniej do zachowania przeciwnika. Skuteczność takiego podejścia jest zależna od typu przeciwnika(patrz punkt 4)). Inny dobór parametrów uczenia, jak również sposobu zachowania robota(np. większy zbiór bardziej różnorodnych akcji) mogłyby mieć znaczący wpływ na skuteczność robota.

Niestety nie udało się zaimplementować uczenia opartego o sieci neuronowe, ale wykorzystanie zwykłego uczenia ze wzmocnieniem daje niezłe rezultaty.

Kod źródłowy projektu dostępny jest pod adresem https://github.com/Lukasz1928/robocode-bot.