

Sztuczna inteligencja

Ćwiczenia 4

(zgodnie z harmonogramem na p4)

Każde zadanie warte jest 1 punkt.

Zadanie 1. W grze Reversi pewne pola są *bezpieczne*, tzn. raz zajęte nie zmienia już w danej grze przynależności. Oczywistym przykładem są pola narożne. Jeżeli jednak o bezpieczeństwie pola będziemy mówić w kontekście konkretnej sytuacji na planszy, to pola narożne są potencjalnie niejednymi bezpiecznymi: przykładowo, jeżeli czarny ma zajęty narożnik, to bezpieczne dla czarnego są dwa przylegające do tego narożnika pola brzegowe.

- a) Jak wykorzystać wiedzę na temat bezpieczeństwa pola w tworzeniu heurystycznej funkcji oceniającej?
- b) Zdefiniuj precyzyjnie warunek bezpieczeństwa pola (w sposób umożliwiający napisanie programu sprawdzającego bezpieczeństwo pola).
- c) Zastanów się, jak w praktyce najlepiej sprawdzać bezpieczeństwo pola, tak żeby funkcja heurystyczna nie zwolniła zbyt wiele. Czy warto definiować jakąś podklasę pól bezpiecznych, które da się szybko wyznaczać?

Zadanie 2. Dla algorytmów Alpha-Beta-Search oraz MCTS odpowiedz na pytania:

- a) W jaki sposób można wykorzystać czas poświęcony na obliczenia najlepszego *poprzedniego* ruchu do obliczenia najlepszego *bieżącego* ruchu (zakładamy, że rozgrywamy tylko jedną partię).
- b) W jaki sposób wykorzystać możliwość równoległego wykonywania kodu w celu poprawy jakości gry?

Zadanie 3. Rozważamy grę „kółko i krzyżyk z buczeniem”. Mamy zwykłą planszę do kółka i krzyżyka, dwóch graczy i serwer gry. Gracze zgłaszają propozycje ruchu do serwera gry (gracz A nie widzi propozycji gracza B i odwrotnie) i serwer albo przyjmuje ten ruch (jeżeli zaproponowane pole jest puste), albo buczy i gracz ma prawo do kolejnej próby (dowolnie wiele razy). Buczenie jest słyszalne dla obu graczy, niedozwolone jest powtarzanie jakiegoś ruchu (tzn. jak przy ruchu przeciwnika usłyszałem buczenie, to mogę założyć, że zdobył on dodatkową wiedzę, a nie że próbuje mnie zmylić). Oprócz tego to normalne kółko i krzyżyk (tzn. celem jest ustawienie 3 swoich znaków w wierszu, kolumnie lub na przekątnej).

Zaproponuj agenta, który grałby w tę grę (możliwie dobrze). Oczywiście pierwszy gracz ma dużą przewagę (dlaczego?), więc zakładamy, że rozgrywka składa się z $2K$ partii, w których gracze rozpoczynają na zmianę.

Zadanie 4. Gry karciane są dobrym przykładem sytuacji, w której nie tylko występuje element losowy (rozdawanie kart), ale również gracz musi podejmować decyzje w sytuacji, w której nie zna w pełni stanu gry (nie wie, co otrzymali inni gracze). Gdy programuje się takiego agenta, często używanym pomysłem jest wielokrotne losowanie możliwego stanu gry (czyli bieżącego układu kart), znajdowanie najlepszego ruchu w tym stanie (używając standardowych metod rozwiązywania gier z jawnym stanem) i ostateczny wybór ruchu, który był najlepszy w największej liczbie losowań.

Wyjaśnij następujące kwestie:

- a) Co może oznaczać: losowanie **możliwego** stanu?
- b) W jakich sytuacjach losowanie z poprzedniego punktu chcielibyśmy wykonywać przypisując stanom niejednakowe prawdopodobieństwa? Co można w ten sposób uzyskać i jakie to rodzi problemy? Uwaga dla osób mało grających w gry karciane: wiele gier ma element licytacji, w której gracze deklarują (często wielokrotnie) jaki wariant przyszłej rozgrywki wydaje im się odpowiedni.
- c) Jaki istotny aspekt gier karcianych jest pomijany w tym podejściu?

Zadanie 5. Rozważmy grę oszust (zob. Cheat_(game) w Wikipedii). Gramy standardową talią 52 kart (asy są najniższymi kartami). Mamy k graczy, którym rozdane są wszystkie karty. Zaczyna rozdający, po czym gracze na zmianę „zagrywają” od 1 do 4 kart (przy czym zagrywają je koszulkami do góry, tak że nie są widoczne dla innych graczy). Celem jest pozbycie się wszystkich kart. Gracz zagrywając kartę (karty) mówi o ich wartości (na przykład mówi: zagrywam dwie siódemki). Można deklarować

jedynie karty o równej wielkości, dodatkowo deklaracja powinna być starsza (bądź równa) deklaracji poprzedniego gracza, czyli na „dwie siódemki” można rzucić „trzy dziewiątki”, ale nie odwrotnie¹. Gracze mogą kłamać odnośnie tego, co rzucili. Po każdej zagrywce jest faza sprawdzania, w której kolejni gracze mogą spasować lub sprawdzić zagrywającego (ta faza albo zawiera $k - 1$ pasów, albo pewną liczbę pasów i pierwsze sprawdzenie). Jeżeli zagrywający kłamał, to zabiera wszystkie karty². Jeżeli zagrywający powiedział prawdę, to karty bierze sprawdzający. W obu przypadkach kolejny gracz kontynuuje rozgrywkę (ponieważ stos jest pusty, może wyrzucić karty o dowolnej wysokości, oczywiście wyrzucając karty może skłamać o ich wartości).

Zaproponuj kilku (co najmniej 2) przykładowych prostych agentów, grających w tę grę.

Zadanie 6. Potestuj playground.tensorflow.org. Odpowiedz na pytania:

- Dla jakich zbiorów danych i jakich cech wystarcza 1 neuron do poprawnej klasyfikacji? (i dlaczego)
- Co dzieje się, gdy dla bardziej złożonych sieci damy zbyt dużą *Learning rate*?
- W którym zadaniu przydają się cechy \sin i \cos ?
- Dla każdego zbioru danych (oprócz spirali) powiedz, jaka najprostsza³ sieć neuronowa korzystająca tylko z cech x_1 i x_2 poprawnie klasyfikuje ten zbiór danych.

Zadanie 7. Rozważamy sieć neuronową z prostą funkcją schodkową w roli σ (równą 1 dla liczb dodatnich, 0 w przeciwnym przypadku). Wejściami do tej sieci będą zera i jedynki, zatem sieć będzie obliczała jakąś funkcję boolowską.

- Podaj sieci neuronowe (złożone z jednego neurona) obliczające $x \vee y$, $x \wedge y$, $\neg x$.
- Podaj sieć neuronową dla $x \text{ xor } y$
- Uzasadnij, że nie jest możliwa sieć z punktu b), która ma tylko 1 neuron

Zadanie 8. Rozważamy takie same sieci, jak w poprzednim zadaniu. Czy za pomocą sieci neuronowych można wyrazić dowolną funkcję boolowską? Jaka jest minimalna liczba warstw, która wystarcza (zakładamy, że neurony mogą mieć dowolną liczbę wejść).

Zadanie 9. Rozważamy procesy MDP. Przyjmujemy, że v_* oraz q_* to odpowiednio wartości stanów i akcji dla optymalnej polityki.

- Pokaż, jak wyrazić v_* za pomocą q_* .
- Pokaż, jak wyrazić q_* za pomocą v_* oraz $p(s', r|s, a)$.
- Pokaż, jak wyrazić π_* (optymalną politykę) za pomocą q_* .
- Pokaż, jak wyrazić π_* za pomocą v_* oraz $p(s', r|s, a)$.

Zadanie 10. ★ Co to jest nadracjonalność (superrationality)? (znajdź odpowiednie informacje w Internecie)

Zadanie 11. ★ Podobnie jak w poprzednim zadaniu, powinieneś posilkować się samodzielnie znalezionymi informacjami. Wyjaśnij, co to jest punkt równowagi Nasha. Opowiedz, na czym polega gra w Dylemat więźnia i jaki jest dla niej punkt równowagi Nasha. Jak twórca agenta grającego w tę grę mógłby wykorzystać następujące fakty:

- Jest wielu graczy, każdy gra w tę grę wiele razy, w różnych parach.
- Każdy gracz przedstawia się przed rozgrywką swoim unikalnym identyfikatorem
- Liczy się sumaryczny wynik wielu rozgrywek.

¹W starszeństwie nie ma znaczenia, ile kart rzucamy, jedynie to, jakie są ich wysokości

²Zasady gry tego nie precyzują, ale możemy przyjąć, że gracz zabiera karty zachowując ich kolejność i jeżeli ma odpowiednio dobrą pamięć, to może po fakcie sprawdzić, kto kłamał, a kto mówił prawdę w podczas budowy tego stosu.

³Mająca najmniej warstw i (w drugiej kolejności) najmniej neuronów.