Dokumentace k projektu

Předmět: Umělá inteligence a strojové učení (SUI)

Projekt: Dicewars AI

Rok: 2020

Autoři:

- Bc. Šamánek Jan (xsaman02)
- Bc. Stojan Radomír (xstoja07)
- Bc. Freyburg Petr (xfreyb00)

Technologie:

- Python3 (testováno s verzí 3.8.5),
- Numpy (testováno s verzí 1.19.4).

Naše vytvořená umělá inteligence se skládá v základu ze 2 částí. Klasifikátor, který určuje vhodnost útoku a algoritmus prohledávání stavového prostoru. Tyto 2 části poté spolupracují pro nalezení nejvhodnějších cest útoku.

1. Klasifikátor

Klasifikátor jsme implementovali pomocí algoritmu *K-nearest neighbours*, kde pro každý nově evaluovaný útok se najde *K* nejbližších ohodnocených útoků. Klasifikátor pak vrací průměrnou hodnotu těchto sousedů.

Prostor pro KNN jsme se rozhodli definovat jako 4-dimenzionální, kde každá dimenze značila jednu sledovanou vlastnost. Jednotlivé vlastnosti byli definované jako:

1. pravděpodobnost dobytí provincie,
2. změna velikosti největšího regionu po dobytí provincie,
3. průměrný počet kostek nepřátel cílové provincie,
4. průměrný počet kostek nepřátel útočící provincie.

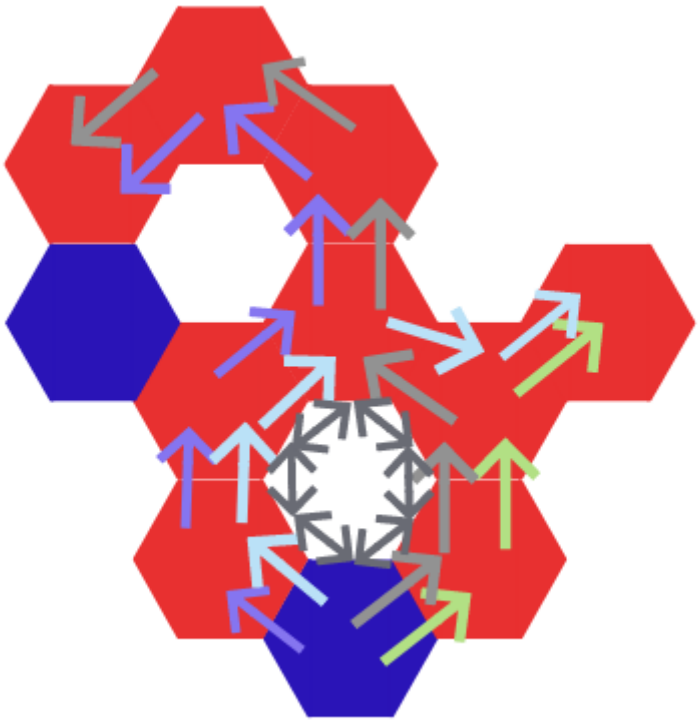
Hodnoty jednotlivých vlastností se normují na hodnoty v rozmezí 0-1 a nejbližší sousedé se počítají pomocí funkce *numpy.linalg.norm()*, která dovoluje jednoduše a efektivně počítat euklidovskou vzdálenost i v n-dimenzionálním prostoru.

Rychlost evaluace jednoho útoku pomocí *numpy.linalg.norm()* byla evaluována na 1 000 000 iteracích a datasetu o 1 000 vektorech o délce 4. Průměrná rychlost nalezení sousedů pro jeden bod byla odhadnuta na 3.10489e-05 sekundy.

2.Prohledávání stavového prostoru

Prohledávání je implementováno pomocí třídy TreeSearch, která obsahuje rekurzivní funkci __expand_node(). Tato funkce vezme danou provincii a nalezne všechny nepřátelské sousední provincie, které má daná provincie kolem sebe. Tyto nepřátelské provincie přidá do pole označující evaluovanou cestu a pro každou nepřátelskou sousední provincii se opět zavolá rekurzivně daná funkce. Funkce ignoruje duplicitní sousedy v cestě a zanořuje se do hloubky maximální útočné síly první provincie v cestě (útočníka) nebo do předem stanoveného omezení nebo pokud už není žádný nepřátelský soused k dané provincii.

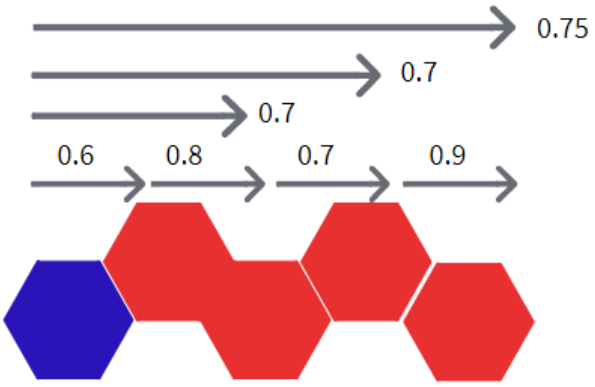
Obr 2.1: znázornění prohledávání cesty. Modré hexagony znázorňují provincie patřící útočníkovi, červené hexagony znázorňují nepřátelské hexagony. Barva šipek znázorňuje zaznamenanou cestu útoku, kterou vrátí funkce __expand_node(). Pro přehlednost jsou ukázány cesty pouze z dolní provincie.



3.Spojení klasifikátoru a prohledávání stavového prostoru

Nalezené cesty z obrázku 2.1 se začnou postupně evaluovat pomocí KNN klasifikátoru. Každý prefix cesty je uložený jako samostatná cesta s indexem vhodnosti útoku, který se počítá jako průměr všech jednotlivých útoků v dané cestě viz obrázek 3.1.

Obr 3.1: znázorňuje vyhodnocování útočné cesty po částech. Každá z kombinovaných šipek uvedených v horní polovině obrázku je uložena jako samostatná cesta společně s indexem vhodnosti.



Takto evaluované cesty se následně seřadí sestupně podle indexů vhodnosti a v takovémto pořadí se začnou jednotlivé útoky vykonávat. Jedná se podstatě o upravenou verzi UCS algoritmu s ohledem na faktor toho, že daná cesta nemusí vyjít.

4.Vykonávání útoků

Útoky se vykonávají postupně podle nejlépe seřazených cest a kontroluje se, zda útok byl úspěšný. Strategie funguje zhruba takto:

1. pro všechny seřazené útoky:
2. vezmi první 2 provincie v cestě (útočník a cíl)
3. pokud útočník patří hráči (minulý útok byl úspěšný nebo je to začátek nové cesty)
4. odmaž z cesty útočníka
5. proveď útok
6. pokud ne
7. koukni jestli danou provincii v minulosti dobyl stejný útočník (ošetření vykonání cesty ze 2 útoků)
8. pokud byl útočník stejný
9. odmaž útočníka a pokračuj v cestě
10. pokud ne
11. smaž celou cestu a začni provádět další

Algoritmus vykoná všechny útoky v seznamu, které byli dostupné a kde index každého jednotlivého útoku byl větší než 0,5.

5.Učení

Algoritmus se učí z odehraných her, a to pomocí náhodného výběru tahů. KNN klasifikátor je na začátku nastavený na daný počet náhodných vektorů (bodů), které jsou uloženy v datasetu. Všechny tyto body mají nastavenou hodnotu na 1 (což je nejvyšší ohodnocení). To znamená, že nově inicializovaný klasifikátor bude vykonávat všechny možné útoky. Následně podle předdefinovaného *learning rate* jsou náhodně vybírány útoky, které algoritmus provádí a tyto útoky se na další kolo vyhodnocují následovně:

- pokud útok nebyl úspěšný, tak bod, pro který připadal tento útok je označený hodnotou 0,
- pokud byl útok úspěšný, ale provincie nevydržela do dalšího tahu -> 0.5,
- pokud byl útok úspěšný a provincie vydržela do dalšího tahu -> 1.

Takto se postupně přidávají ohodnocené body do datasetu, který se z původních 100 bodů zvětší například na 2 000 bodů.

6.Rychlost

Rychlost algoritmu značně závisí na počtu možných cest útoku. Algoritmus si všechny cesty vyhodnotí na začátku svého tahu a po zbytek tahu už pouze realizuje útoky. Maximální naměřený čas celého tahu byl okolo 4 sekund, ale průměrný čas na kolo (provedení všech chtěných útoků), který byl evaluován na 100 duelech byl změřen na 0.2 sekundy (testováno na intel i5 8265U).

7.Výsledky

Výsledky byly vyhodnoceny na turnaji o 800 hrách 4 hráčů, kde naše inteligence dopadla s *winratem* 37,19 % a dále na 400 duelech, kde výsledek naší AI byl 57,89 %, se kterým se společně s dt.wpm_c umístil na prvním místě.

8.Vypozorovaná zlepšení

Výsledek natrénování hodně závisí na způsobu trénování. V případě trénování na hrách více hráčů se s velkou šancí vytvoří skeptické AI, které nebude příliš útočit. Často se pak také stávalo, že AI mělo dobýt poslední provincii v souboji 8vs8 kostek a AI se rozhodla nezaútočit, protože šance nebyla dostatečná. To mělo za následek zacyklení a následné ukončení hry z důvodu nečinnosti.

Nejvíc se nám osvědčil trénink proti skutečnému hráči s vysokým *learning rate* a následné lehké dotrénování na hře více hráčů.

9.Potencionální zlepšení

Určitě lepší zpětná propagace výsledků z předchozího kola do datasetu. Propagace je příliš závislá na samotné pravděpodobnosti úspěchu dobytí a nevěnuje příliš pozornosti spojování vlastních regionů a rozdělování nepřátelských regionů.