Maturitní práce z informatiky

Piškvorky s počítačovým protivníkem

Tadeáš Báča, 08.B

Praha, 14. dubna 2023

Vedoucí práce: Vojtěch Horký
Gymnázium, Praha 6, Nad Alejí 1952

Zadání maturitní práce z informatiky

Jméno žáka: Tadeáš Báča
Název práce: Piškvorky s počítačovým protivníkem
Termín odevzdání: 14. dubna 2023
Jméno vedoucího: Vojtěch Horký

Rozpis podmínek na jednotlivé části práce

Obecné podmínky odevzdání popisuje příloha společná pro všechny maturanty.

Obsah práce

Naprogramujte hru piškvorky s počítačovým protivníkem.

Funkce

Aplikace nabídne klasickou hru piškvorek s možností hry proti počítačovému protivníkovi.

Rozhraní

Aplikace bude fungovat v grafickém režimu.

Počítačoví protivníci

Těžiště práce je v tvorbě zdatného protivníka řízeného počítačem.

Ten bude založen na umělé inteligenci: implementace bude v sobě obsahovat neuronovou síť, která bude předem natrénována. Toto „naučení se“ bude založeno na některém z obvyklých způsobů, jako je třeba hodnocení náhodných tahů (zda vedou k výhře či nikoliv).

Specifické pokyny pro vypracování

Řešitel využije existující knihovny pro vytvoření neuronové sítě (nebo jiného prostředku UI).

Toto zadání práce schválil dle příslušné maturitní vyhlášky a žákovi zadal ředitel školy.

V Praze dne 9. prosince 2022 .....
podpis ředitele a razítko

Prohlášení

Prohlašuji, že svou práci na téma *Piškvorky s počítačovým protivníkem* jsem vypracoval/a samostatně pod vedením a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce. Dále prohlašuji, že tištěná i elektronická verze práce jsou shodné a nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a změně některých zákonů (autorský zákon) v platném znění.

Praha, 14. dubna 2023 Tadeáš Báča .....

Anotace

Cílem této práce je vytvořit počítačového protivníka ve hře piškvorky. Počítačový protivník bude založen na umělé inteligenci a strojovém učení. Tato práce implementuje algoritmy *Deep Q-Learing* a *AlphaZero*.

Klíčová slova

piškvorky, strojové učení, neuronové sítě, učení posilováním, AlphaZero

Anotation

The aim of this work is to create a computer opponent for the game of Tic Tac Toe. The computer opponent will be based on artificial intelligence and machine learning. This work implements the *Deep Q-Learing* and *AlphaZero* algorithms.

Keywords

Tic Tac Toe, machine learning, neural networks, reinforcment learning, AlphaZero

Obsah

1 Úvod **5**

2 Uživatelská dokumentace **6**
2.1 Návod k instalaci 6
2.2 Používání 6

3 Teorie **7**
3.1 Neuronové sítě 7
3.1.1 Neuron 7
3.1.2 Konvoluční neuronová síť 7
3.2 Učení posilováním 7
3.2.1 Deep Q-Learning 7
3.2.2 AlphaZero 9

4 Závěr **10**

5 Zdroje **12**

Úvod

Jako téma své maturitní práce jsem si vybral umělou inteligenci, protože s ní už mám zkušenosti a chtěl jsem rozšířit své znalosti. Dlouho mě zajímalo učení posilováním, ale nevěděl jsem kde začít. Hra piškvorky má jednoduchá pravidla, ale skrývá v sobě značnou komplexitu, a právě proto je dle mého názoru tento typ hry ideální na experimentování a zkoušení algoritmů učení posilováním. Byl bych rád kdyby tato práce fungovala jako vstupní bod pro ostatní nadšence zkusit si zaexperimentovat s nyní populárními postupy.

Uživatelská dokumentace

2.1 Návod k instalaci

1. Stáhněte si .zip soubor na `https://tad34s.github.io/#o_projektu`.
2. Soubor extrahujte.
3. Ze stránky: `https://github.com/tad34s/piskvorky-rl-maturitni-prace/tree/main/bot/Players/Alpha_Zero/NNs_preset` si stáhněte jeden z předem natrénovaných modelů.
4. Vložte model do složky *active_model*
5. Spusťte .exe soubor

2.2 Používání

Kliknutím zahrajete tah, nebo když je hra u konce, odstartujete novou. Proti Vám bude hrát AlphaZero model s parametry, které jsou načteny ze souboru v *active_model*. Pokud chcete zkusit jiný model, vyměňte soubor.

Teorie

V této kapitole se budu věnovat tomu, jak fungují tyto algoritmy a jak jsem je sám naimplementoval.

3.1 Neuronové sítě

Jak název napovídá, z jistého pohledu se jedná o matematické napodobení struktury mozku. Stejně jako u něj je v neuronové síti základní stavební jednotkou *neuron*. Tyto neurony jsou seřazeny do vrstev. Neuron získává svoje vstupy ze všech výstupů neuronů v předešlé vrstvě.

3.1.1 Neuron

Neuron je zde matematická funkce, která má předem specifikovaných N vstupů, $N+1$ parametrů, aktivační funkci a pouze jeden výstup. Pro každý vstup má neuron parametr w_n, kterému se říká váha. Vstup vynásobí váhou a potom je všechny sečte. Nakonec přičte poslední parametr b, neboli práh. Tato suma se poté vloží do aktivační funkce.

3.1.2 Konvoluční neuronová síť

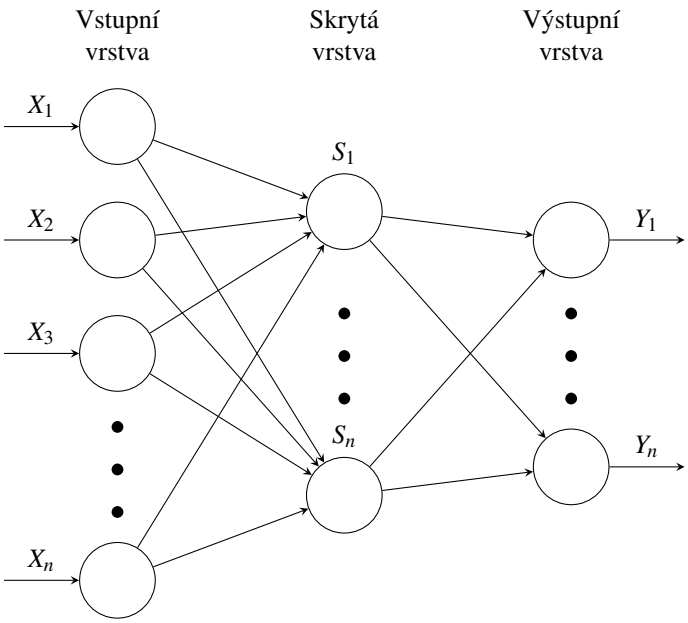
Konvoluční neuronová síť používá konvoluční vrstvu. Tato vrstva nepoužívá neurony, ale tzv. "okénka". Okénka procházejí vstup a provádějí na něm stejnou operaci. Často se používají pří práci s obrázky.

3.2 Učení posilováním

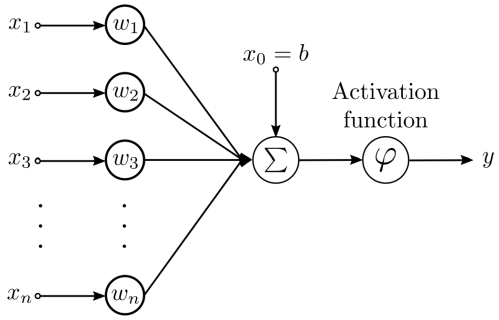
Učení posilováním je druh strojového učení, kde máme agenta nacházejícího se v prostředí, ve kterém může získávat odměnu R. Naším úkolem je najít pravidlo π, podle kterého se bude agent řídit a které maximalizuje získanou odměnu. Tedy $\pi(s) = a$, kde a je akce.

3.2.1 Deep Q-Learning

Deep Q-Learning je jeden ze způsobů, jak docílit optimálního pravidla π. Základem tohoto algoritmu je Q funkce. Tato funkce ohodnotí ve stavu s akci a hodnotou, která odpovídá všem odměnám, kterou agent v



Obrázek 3.1: Neuronová sít



Obrázek 3.2: Neuron

budoucnu po udělání této akce dostane. Pravidlo π je jednoduše akce s nejvyšší Q hodnotou. Jelikož jsou odměny, které agent dostane dřív trochu hodnotnější, vynásobíme budoucí hodnoty zlevněním γ.

$$Q(s_t, a) = R(s_{t+1}) + \gamma R(s_{t+2}) + \gamma^2 R(s_{t+3})...$$

To si můžeme upravit na:

$$Q(s_t, a_t) = R(s_{t+1}) + \gamma \max Q(s_{t+1})$$

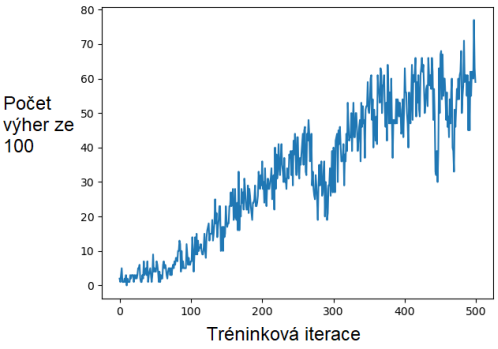
Funkce Q je nahrazena neuronovou sítí, ve které se počet výstupních neuronů rovná počtu všech možných akcí. Tato sít se nazývá DQN neboli Deep Q-Network, protože má často hodně vrstev. Učení probíhá tak, že hráč hraje hry a zapamatovává si stav s_t, akci a_t a stav po něm následující s_{t+1}, poté si z výše uvedené rovnice vypočte novou hodnotu.

3.2.2 AlphaZero

AlphaZero je algoritmus vyvinutý společností DeepMind, který byl úspěšný ve hraní šach, go a podobně. Má dva základní komponenty - Monte Carlo Tree Search a neuronovou sít. Monte Carlo Tree Search je heuristický algoritmus, který podle daného pravidla prochází strom všech možných tahů. Poté vrací nejlepší tah, který našel. V algoritmu AlphaZero je tato heuristika zaměněná za neuronovou sít. Tato neuronová síť ma dvě hlavy, jedna určuje, který stav se má prohledat jako další a druhá jaká je hodnota stavu, pokud není konečný. AlphaZero se učí hraním sám se sebou. Hodnoty na učení bere z prohledávání stromu. Čím více ho prohledá, tím více se zlepší.

Závěr

V této práci se mi podařilo úspěšně implementovat řadu algoritmů, ať už nějaké jednodušší na trénování umělé inteligence, tak i samotné hráče s neuronovou sítí. Modely se úspěšně učily a zlepšovaly. Bohužel se ale zatím nepodařilo dostat neuronovou sít na lidskou úroveň. Kvůli mým omezeným výpočetním možnostem trvá trénování modelu dlouho. Zlepšení umělé inteligence se také rychle zastropuje. Nemyslím si, že by to zlepšila větší neuronová sít. Možná by se jí podařilo hrát proti konkrétnímu soupeři lépe, špatně by ale generalizovala, jelikož se do ní často více promítne menší změna vstupů. Nejlépe dopadl model AlphaZero. Při trénování jsem často zastihl opravdu zajímavé a promyšlené hry, ale na to, aby se promítly do her s lidským protihráčem bude potřeba více trénování.



Obrázek 4.1: Hry proti hráči, který pouze útočí

```
One move:  1.0356640815734863 seconds
________
________
____O___
___X_X__
____X___
_____O__
________
________

One move:  1.1556251049041748 seconds
________
___O____
____O___
___X_X__
____X___
_____O__
________
________

One move:  1.0676534175872803 seconds
________
___O____
____O___
___XXX__
____X___
_____O__
________
________
```

Obrázek 4.2: Příklad hry v konzoli

Zdroje

- BASOVNÍK, Martin. Hra Gomoku [online]. Brno, 2011 [cit. 2023-04-14]. Dostupné z: http://hdl.handle.net/11012/55692. Bakalářská práce. Vysoké učení technické v Brně. Fakulta informačních technologií. Ústav inteligentních systémů. Vedoucí práce Jan Kněžík.
- MUZIKA, Martin. Application of Game Theoretic Algorithms to Gomoku. Dostupné z: http://hdl.handle.net/10467/70078. Bakalářská práce. České vysoké technické učení v Praze. Fakulta elektrotechnická. Vedoucí práce Jiří Čermák.
- PyTorch dokumentace, Dostupna na: https://pytorch.org/docs/stable/index.html
- NumPy dokumentace, Dostupna na: https://numpy.org/doc/stable/

Seznam obrázků

3.1 Neuronová sít 8
3.2 Neuron 8

4.1 Hry proti hráči, který pouze útočí 11
4.2 Příklad hry v konzoli 11