

Effect of unsupervised pretraining in deep reinforcement learning

Michal Chovanec, michal.nand@gmail.com

Peter Šarafín, peter.sarafin@gmail.com

Keywords: reinforcement learning, function approximation, SARSA, Q-learning, neural network, unsupervised training

Abstract:

Článok sa zaoberá vplyvom unsupervised learning na celkový priebeh reinforcement learning tréningu. Ako aproximátor funkcie ohodnotení je zvolená hlboká neurónová sieť. Overovali sme hypotézu, že unsupervised predtrénovaná sieť sa v režime supervised učí rýchlejšie. Zároveň sa ale očakáva, že obe siete po dostatočnom počte iterácií učenia poskytnú rovnaké výsledky. Dalšia testovaná hypotéza je vplyv riedkosti váh na kvalitu výsledku - dnešné technické prostriedky sú navrhnuté na výpočty hustých sietí **TODO citovať**, v budúcnosti, by však hárdiverovo akcelerované riedke siete mohli priniesť výrazný nárast výkonu **TODO citovať**. Dôležité je preto overiť vplyv riedkych váh na výsledok.

1 Introduction

Reinforcement learning umožňuje tréňovať agenta v Markovových rozhodovacích procesoch tak aby maximalizoval dosiahnutú odmenu. Tradične sú používané algoritmy Q-learning **TODO citovať** a SARSA **TODO citovať**. V tomto článku je použitý algoritmus SARSA, popísaný ako

$$Q'(s, a) = (1 - \alpha)Q(s, a) + \alpha(R(s, a) + \gamma Q(s', a')) \quad (1)$$

kde

$Q(s, a)$ je ohodnotenie akcie a v stave s ,

$R(s, a)$ je odmena za vykonanie akcie a v stave s ,

$\alpha \in (0, 1)$ je rýchlosť učenia,

$\gamma \in [0, 1]$ je discount factor,

a $Q'(s, a)$ je nová hodnota ohodnotenia.

Pre malé stavové priestory (väčšinou len učebnicové príklady) je možné použiť tabuľku na uloženie Q hodnôt. Pre veľké stavové priestory - je nevyhnutná aproximácia Q hodnôt. Často používané sú lineárna kombinácia bázových funkcií **TODO citovať** alebo dopredná neurónová sieť **TODO citovať**.

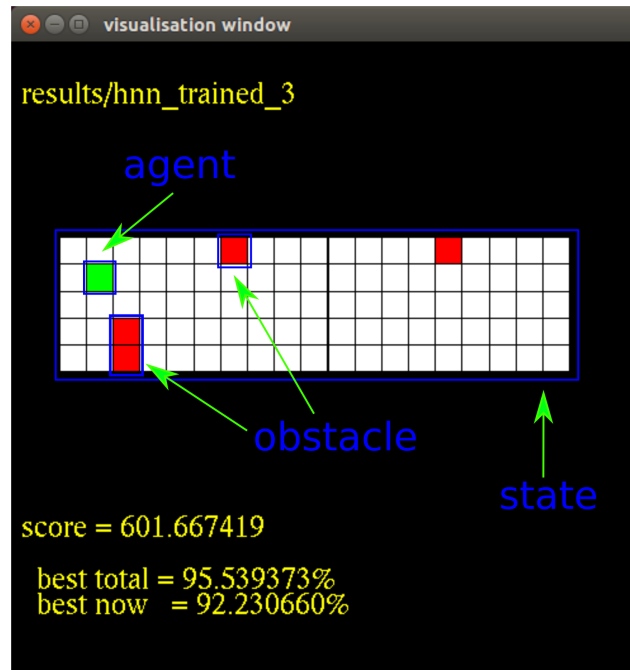
Rekurentná povaha rovnice 1 spôsobuje že učiaci proces je neefektívny a pomalý.

Naša myšlienka je zameraná na unsupervised predtrénovanie siete a až po tom tréňovať použitím SARSA algoritmu.

2 Popis experimentu

Agent bol umiestnený do hry znázornenej na obrázku 1. Úlohou je preskakovať a podliezať červené prekážky, agent má teda na výber dve akcie. Prekážky sa generujú náhodne, a postup je sprava do ľava. Za úspešné vyhnutie sa kolízií získa agent odmenu $+0.3$, za neúspech -1.0 . Toto je nevyvážené, je dôležité, pre kvalitatívne overenie algoritmu, aby náhodnými akciami nikdy nedosiahol kladné skóre.

Stav sú jednotlivé políčka hry, v tomto prípade 5×19 (95 prvkový vektor). Hodnota 0 znamená voľné (biela), hodnota 1 reprezentuje agenta (zelená), a hodnota -1 je zvolená pre prekážku (červená).



Obr. 1: Testing arcade game

V hre sa sleduje niekoľko veličín. Celkové skóre je dané súčtom jednotlivých odmien. Ďalej sme zaviedli ďalšie dve pomocné veličiny - `best_total` a `best_now`.

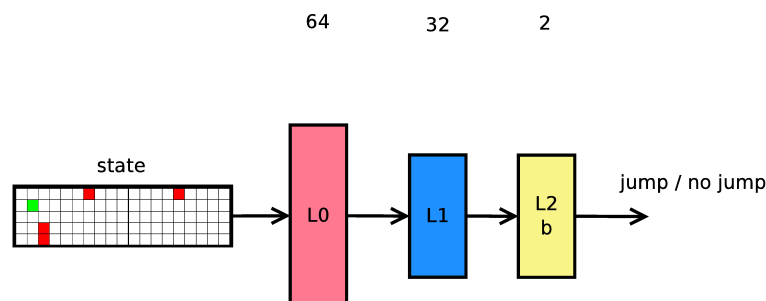
Veličina `best_total` ukazuje, koľko percent času volil agent správnu akciu - vyhol sa kolízií, od začiatku behu hry ¹.

Veličina `best_now` ukazuje voľbu najlepšej akcie len za posledné obdobie - je použitý priemer s exponenciálnym zabúdaním.

2.1 Topológia siete

Pre porovnanie sme zvolili dve siete. Vstupom je stav hry 5×19 (95 prvkový vektor), výstupom je hodnota $Q(s)$ pre každú akciu - skočiť / neskočiť.

Prvá sieť je dopredná sieť (FNN) s dvoma skrytými vrstvami, topológia je na obrázku 2. Čísla nad vrstvami sú počty neurónov. Skryté vrstvy majú aktivačnú funkciu ReLU, výstupná vrstva lineárnu aktivačnú funkciu.



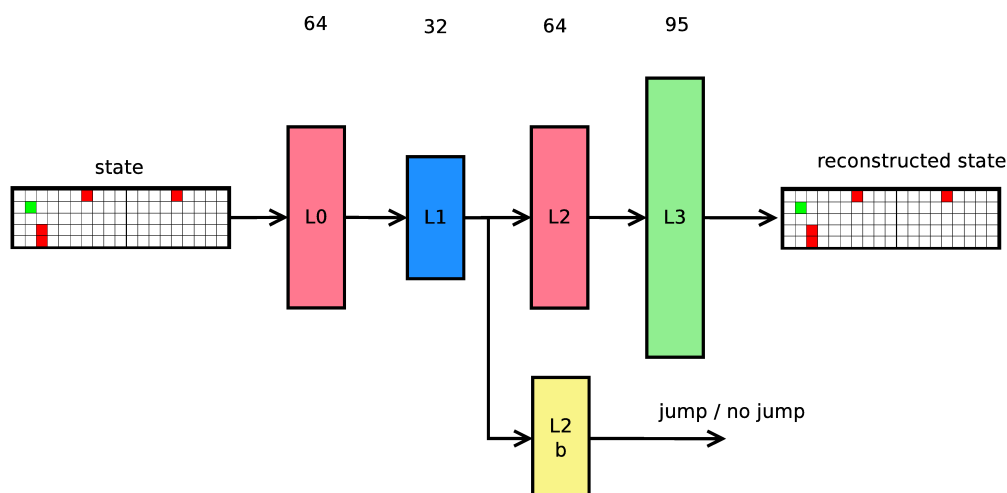
Obr. 2: Supervised network architecture (FNN)

Druhá sieť je tvorená hlbokým autoenkóderom a doprednou sieťou (AE + FNN). Skryté vrstvy majú aktivačnú funkciu ReLU, výstupná vrstva lineárnu aktivačnú funkciu. Jej topológia je na obrázku 3. Vo fáze unsupervised sa trénuje autoenkóder - z 95 prvkového vstupu, postupne extrahuje príznaky, až na počet 32. Následne ďalšie dve vrstvy urobia rekonštrukciu vstupu. Ako učenie sme zvolili metódu stacked autoencoder **TODO citovať**:

¹Orientačne sme zistili, že ľudský hráč je schopný hrať v intervale od 91% do 93%.

najskôr sa urobí bypass medzi L1 a L2 vrstvami a trénujú sa len vrstvy L0 a L3, po zvolenom počte iterácií sa pripoja vrstvy L1 a L2 a autoenkóder sa dotrénuje.

Po natrénovaní autoenkódera sa odpoja vrstvy L2 a L3. Pripojením vrstvy L2b sa sieť už trénuje supervised, podľa rovnice 1.



Obr. 3: Unsupervised + Supervised network architecture (AE+FNN)

2.2 Parametre experimentu

Experiment má mnoho parametrov, a bolo potrebné ich zafixovať - dnes nie je možné analyticky stanoviť ich optimálne hodnoty. Hyperparametre sietí boli experimentálne zvolené tak aby nedochádzalo k divergenciám váh. Úmerne tomu boli upravené počty iterácií učenia.

Veľkosť batch bol zvolený na 1000 po sebe idúcich stavov. Každý batch mal pri tréningu 10 behov (epoch). Parameter SARSA algoritmu γ bol zvolený 0.8, parameter α je pri použití neurónovej siete ukrytý v learning rate parametre siete. Zvolená bola ϵ greedy stratégia voľby akcií. Voľba horšej akcie bola počas tréningu 0.1, počas testovania 0.0 - vyberala sa len najlepšia akcia. Siete boli učené algoritmom backpropagation. Riedkosť váh sa dosahovala pomocou L1 normy. Gradientová metóda učenia váh potom prejde na vzťah

$$\Delta w = \eta E x \frac{df(y)}{dw} - \lambda \text{sgn}(w) \quad (2)$$

kde

E je chyba neurónu,

x je vstup do neurónu,

y je výstup neurónu,

f je aktivačná funkcia,

η je learning rate ,

λ je parameter riedkosti váh.

Hyperparametre siete sú zhrnuté v tabuľke 1.

Tabuľka 1: Hyperparametre siete

	FNN sparse	FNN no sparse	AE+FNN sparse	AE+FNN no sparse
unsupervised iterations	0	0	100000	100000
supervised iterations	200000	200000	200000	200000
iterations per slice	0	0	50000	50000
learning rate	0.0005	0.0005	0.0005	0.0005
init weight range	0.1	0.1	0.1	0.1
dropout	0	0	0	0
lambda	0.00000001	0	0.00000001	0

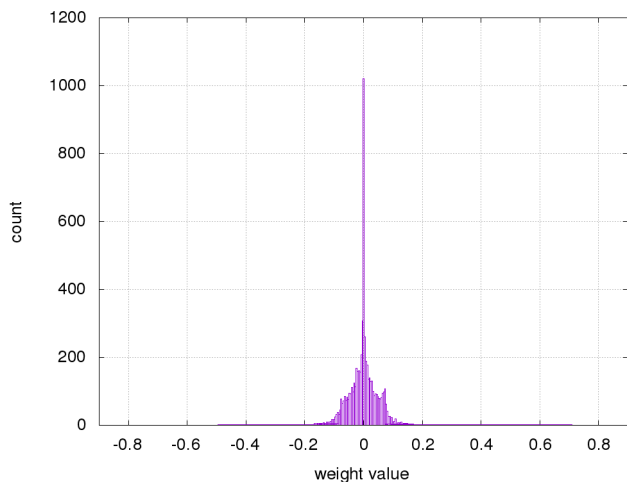
Menili sa parametre **unsupervised iterations**, **iterations per slice** a **lambda**. Parameter iterations per slice je počet iterácií na jednu úroveň stacked autoenkódera.

Sputené boli teda 4 rôzne experimenty, každý 5 krát, aby sa vylúčil vplyv počiatočných hodnôt váh a dosiahol sa štatisticky vierohodný výsledok.

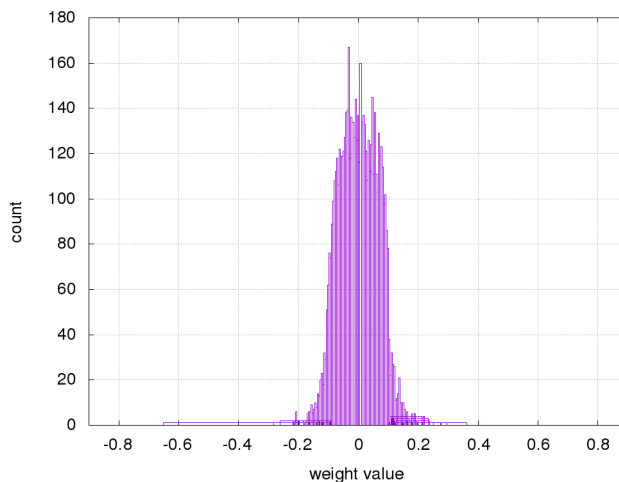
3 Výsledky experimentov

3.1 Riedkosť váh

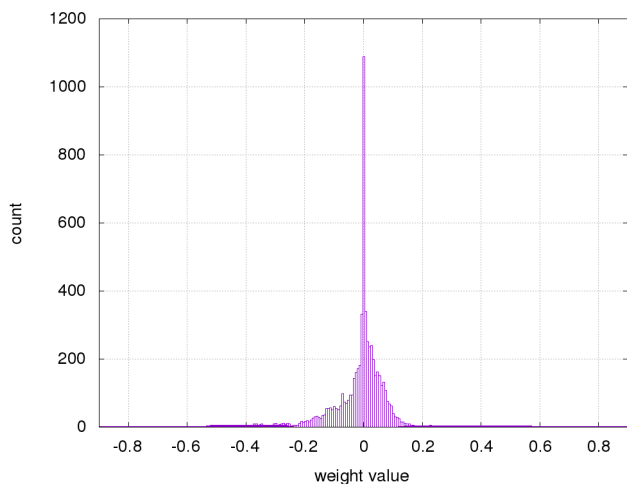
Počas tréningu sa postupne váhy ustália na potrebné hodnoty. Pomocou histogramov je možné overiť riedkosť váh, v závislosti od parametra λ . Pre prehľadnosť sú znázornené len vlastnosti váh pre prvú vrstvu siete. Histogramy na obrázkoch 4 a 5 porovnávajú početnosť váh pre doprednú sieť. Pre riedku sieť je 16% váh nulových, pre neriedku 1.67%. Podobné výsledky sú pre kombináciu autoenkódera a doprednej siete - obrázky 6 a 7 S percentuálnym zastúpením 17.7% pre riedke váhy a 3.3% pre neriedke váhy. Ilustračne ešte uvádzame charakter riedkych váh pre každý zo 64 neurónov, usporiadaných do matice 5x19 - tak aby zodpovedali vizuálnej interpretácii prostredia. Obrázky 8, 9.



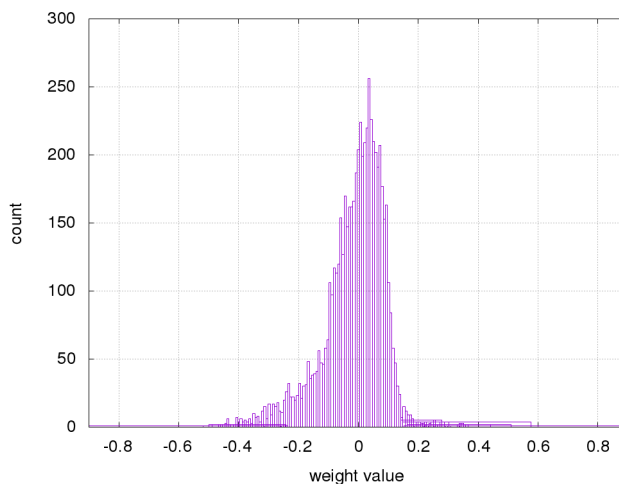
Obr. 4: FNN sparse weights histogram



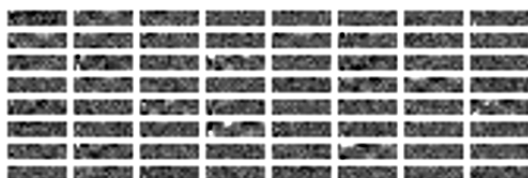
Obr. 5: FNN no sparse weights histogram



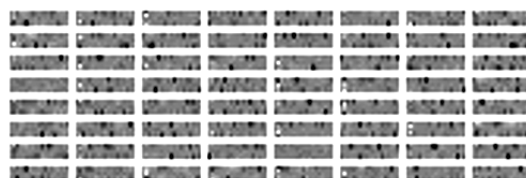
Obr. 6: AE+FNN sparse weights histogram



Obr. 7: AE+FNN no sparse weights histogram



Obr. 8: FNN sparse weights visualisation



Obr. 9: AE+FNN sparse weights visualisation

3.2 Dosiahnuté skóre

TODO : viac to rozpisat

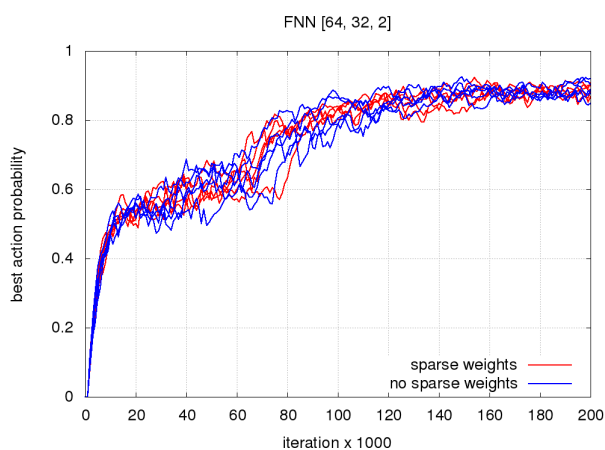
Po 200 000 supervised učiacich iteráciach sa spustilo 50 000 testovacích iterácií. Zhrnuté dosiahnuté skóre je uvedené v tabuľke 2.

Tabuľka 2: Summary results

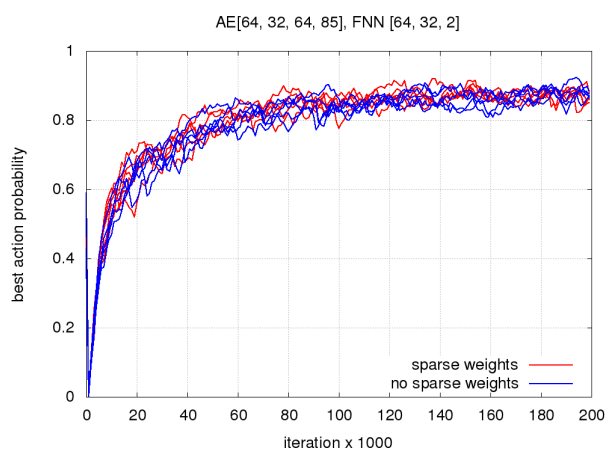
	average score	best score	worst score	average best action probability [%]
FNN sparse weights	960.58	994.97	922.64	95.32
FNN nosparse weights	945.04	995.64	878.31	93.29
AE+FNN sparse weights	914.5	947.64	875.31	93.4
AE+FNN no sparse weights	908.58	954.31	780.32	93.12

Testovanie hypotézy o rýchlosti učenia zobrazujú grafy na obrázkoch 10, 11, 12.

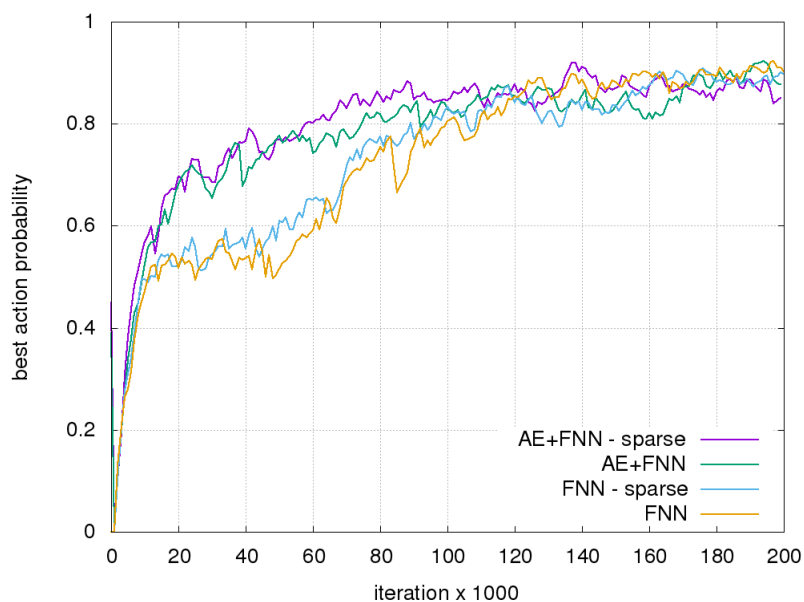
Z grafu obrázka 12 je zrejmé, že predtrénovaná sieť sa učí rýchlejšie. Po čase podľa očakávania, obe siete dosiahnu rovnaký výsledok. Hypotéza sa teda potvrdila. Pre prehľadnosť, sú v grafe znázornené len 4 vybrané siete. Táto redukcia je možná vďaka malým rozptylom hodnôt všetkých testovaných sietí - priebehy pre všetkých 10 sietí (5 riedkych a 5 neriedkych) je možné vidieť na obrázkoch 10 pre doprednú sieť a 11 pre predtrénovanú sieť. Rovnako nie je viditeľný kvalitatívny rozdiel medzi riedkymi a neriedkými váhami - optimalizáciou harvéru na riedke výpočty by tak mohlo byť výrazne urýchlené.



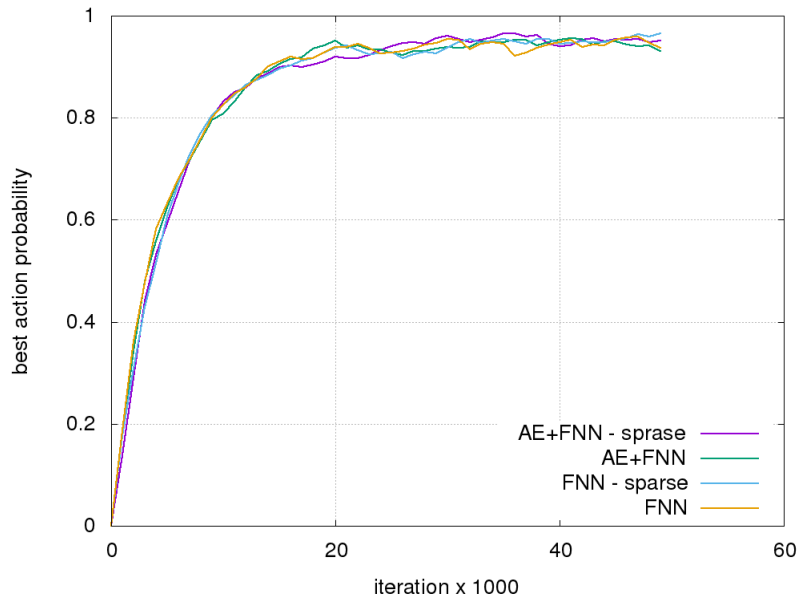
Obr. 10: FNN progress comparison



Obr. 11: AE+FNN progress comparison



Obr. 12: Training progress comparison



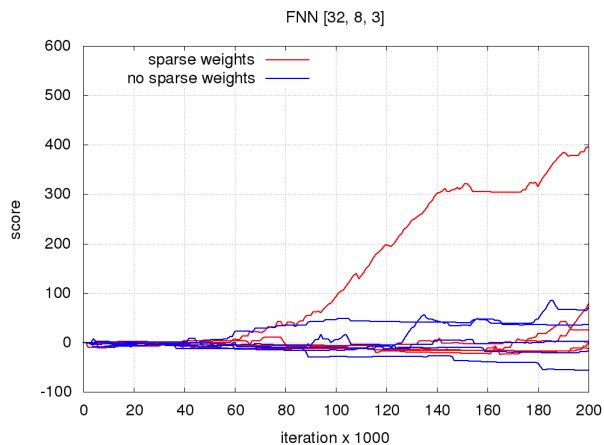
Obr. 13: Testing progress comparison

4 Additional experiment - snake

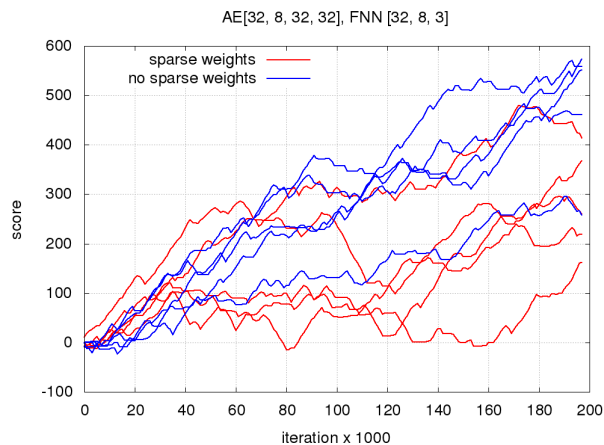
Pozitívne výsledky uvedeného experimentu nás viedli urobiť ešte jeden, doplnkový experiment. Použili sme klasickú hru had - agent má za úlohu zbierať jedlo, obrázok 14. Za každé zjedenie je odmena $+0.1$, za vypadnutie z mapy -1.0 . Agent - had, má na hlave 32 senzorov, ktoré ho informujú o vzdialenosti k jedlu. Sensory sú rozmiestnené v polkruhu, tak aby rovnomerne pokrývali zorné pole. Agent má na výber tri akcie - vpred, vľavo, vpravo. Had je modelovaný ako diferenciálny podvozok **TODO citovať** so zotrvačnosťou. Uvedené akcie teda generujú dve sily, ktoré menia polomer zatáčania, tak ako je uvedené na obrázku ?? **TODO obrazok had senzory + sily**.



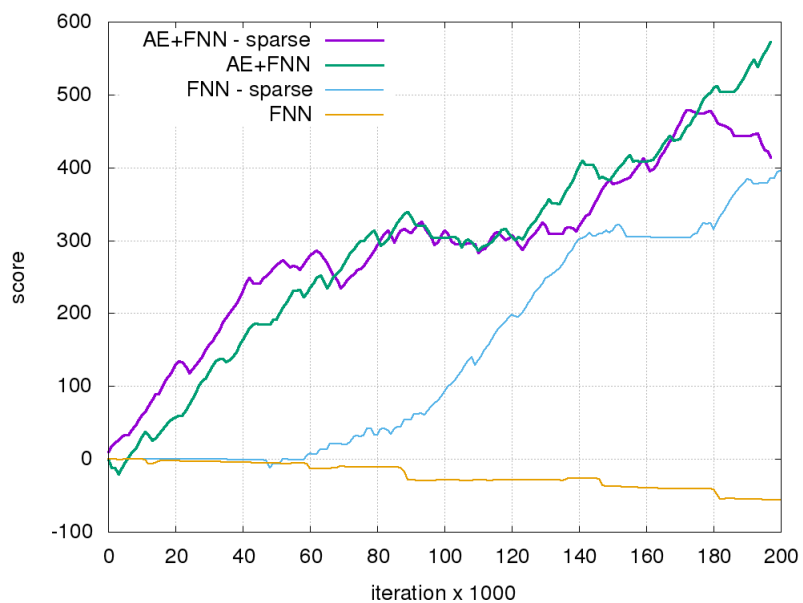
Obr. 14: Testing snake game



Obr. 15: FNN score progress comparison



Obr. 16: AE+FNN score progress comparison



Obr. 17: Training worms score progress

5 Conclusion

Simulačný nástroj bol napísaný v C++, neurónové siete sú počítané pomocou NVIDIA Cuda technológie. Je možné voliť medzi počítaním na CPU a GPU - výsledné zlepšenie závisí od veľkosti siete. Beh experimentov bol asi 60 minút.

Literatúra

- [1] M.J.Flynn, P.Kanerva, and N.Bhadkamkar, 1989, Sparse Distributed Memory: Principles and Operation <http://i.stanford.edu/pub/cstr/reports/csl/tr/89/400/CSL-TR-89-400.pdf>
- [2] David Rogers, 1988, NASA, KANERVA'S SPARSE DISTRIBUTED MEMORY: AN ASSO-CIATIVE MEMORY ALGORITHM WELL SUITED TO THE CONNECTION MACHINE <https://pdfs.semanticscholar.org/9288/bb551f000348f800ff40d0fdb3fd74c410ef.pdf>
- [3] J. S. Albus, 1975, Data Storage in Cerebellar Model Articulation Controller <https://www.cs.cmu.edu/afs/cs/academic/class/15883-f13/readings/albus-1975.pdf>
- [4] Olshausen and Field (1997): Sparse coding with an overcomplete basis set
- [5] Olshausen BA, Field DJ (2004) : Sparse coding of sensory inputs. Current Opinion in Neurobiology, 14, 481-487
- [6] Mushroom body, locust (Laurent)
- [7] HVC, zebra finch (Fee)
- [8] Auditory cortex, mouse (DeWeese & Zador)
- [9] Hippocampus, ratprimate(Thompson & Best; Skaggs)
- [10] Motor cortex, rabbit (Swadlow)
- [11] Visual cortex, monkeycat (Vinje & Gallant)
- [12] Inferotemporal cortex, human (Fried & Koch)