Rešavanje problema Lunar Lander-a pomoću DQN-a

Bane Gerić

Softversko inženjerstvo i informacione tehnologije, Fakultet tehničkih nauka – Novi Sad
Mentor: Branislav Anđelić

Uvod

Reinforcement learning se pojavio krajem prošlog vijeka, medjutim nije zaživio sve do 2013. godine kada je *DeepMind* izbacio svoj model koji je bio sposoban da istrenira većinu *Atari* igara. Oni su koristili *RL* koji se zasniva na principu maksimiziranja nagrade, odnosno minimiziranja kazne, sa dodatkom neurosnke mreže i tako kreirali *DQN* koji je danas u širokoj upotrebi.

Ovaj projekat predstvalja primjenu DQN-a zajedno sa *replay experience* metodom na problem upravljanja raketom, tzv. *Lunar Lander*.

Opis problema

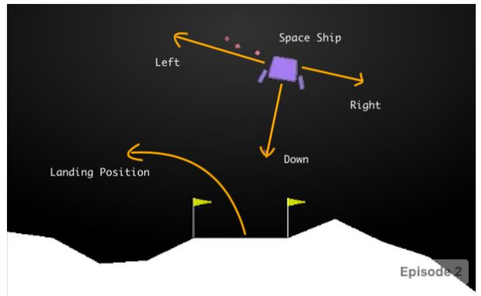
Okruženje daje stanje koje je reprezentovano nizom od 8 elemenata: x i y kooridnate agenta, vertikalna i horizontalna brzina, ugao, ugaona brzina i 2 indikatora koja pokazuju da li je lijeva, odnosno desna noga dodirnula tlo. Na osnovu ovih podataka potrebno je izvršiti najpogodniju od 4 akcije. Akcije predstavljaju koji motor je uključen i mogu biti:

- Ništa
- Lijevi motor
- Glavni motor
- Desni motor

Metodologija

Korišten je metod *Reinforcement Learning (RL)* sa dodatkom neuronske mreže. Ovaj spoj se naziva *Deep Q-learning(Deep Q Network)*. Metod radi na principu da se trenutno stanje koje je dobijeno od okruženja proslijedi neurosnkoj mreži, a izlaz iz neuronske mreže će biti Q vrijednosti. Potom je potrebno izabrati akciju koja odgovara najvećoj Q vrijednosti i izvršiti je.

Da bi se ubrzao i osigurao napredak agenta kroz epizode korišten je *experience replay* koji predstavlja memoriju agenta pomoću koje on pamti poteze koje je uradio u početnim epizodama i onda koristi te informacije kako bi poboljšao svoje kretanje. Pored ovog korištena je i dodatna neuronska mreža koja je služila da „ublaži“ dobijene Q vrijednosti, jer je ponašanje agenta, naročito u ranim fazama, poprilično slučajno.



Izgled okruženja Lunar Lander-a

Najbolje obučeni model je trenutno model koji ima dodat parametar *TAU* koji služi da ublaži ažuriranje sekundarne neuronske mreže. Ovaj parametar je postavljen na 0.001. Pored toga ima 2 *hidden layer-a* sa po 64 jedinice koji koriste *ReLU* kao funkciju aktivacije. Ima *BUFFER_SIZE* podešen na 100.000 i ograničeno vrijeme od 1000 frejmova po epizodi.

Testiranje i rezultati

Zajednički parametri koji nisu mijenjani su: ALPHA=0.0005, BATCH_SIZE=64, GAMMA=0.99, epsilon=1, EPSILON_END=0.01. U modelu 3 je dodata pomoćna neuronska mreža te je iz tog razloga on bio znatno superiorniji u odnosu na druga 2 modela.

	100 epizoda	500 epizoda	1000 epizoda	1230 epizoda	BUFFER SIZE	MAX TIME	EPSILON DECAY	TAU
Model 1	-343.72	-245.55	-143.22	/	10.000	inf	0.996	/
Model 2	-435.50	-170.39	-138.61	/	12.000	inf	0.996	/
Model 3	-245.64	-86.18	-9.31	249.86*	100.000	1000	0.995	0.001

Zaključak

Iz priloženih rezultata se jasno vidi da je model 3 ostvario najbolje performanse. Tome je najviše doprinijela pomoćna neuronska mreža, ali takodje mali dio zasluga ide veličini *buffer-a*. Pošto je uočen veliki napredak kod modela 3 on je istraniran jos 230 epizoda a rezultat u tabeli predstavlja prosjek poena od 1220. do 1230. epizode.
Možemo zaključiti da se DQN veoma dobro pokazao u rešavanju ovog problema, jer je agnet uspješno savladao igru. Jedina mana jeste ta što je za treniranje modela 3 bilo potrebno preko 20 sati, ali to ide na račun hardvera.

Reference

- https://www.youtube.com/watch?v=qfovbG84EBg&list=PLQVvvaa0QuDezJFIOU5wDdfy4e9vdnx-7&index=6
- https://github.com/rogerxcn/lunar_lander_project/blob/master/dqn_agent_keras.py
- https://www.youtube.com/watch?v=5fHngyN8Qhw
- https://goodboychan.github.io/python/reinforcement_learning/pytorch/udacity/2021/05/07/DQN-LunarLander.html?fbclid=IwAR28UD-EwkYbIp0qEmuEiqxtdbjZiye0WwUhiUZu9lvfBDAef-eJo1rwCRc