Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 004.9
DOI: https://doi.org/10.24867/06BE06Njegovanovic

NAVIGACIJA U TRODIMENZIONALNOM OKRUŽENJU *Obstacle Tower* UPOTREBOM UČENJA USLOVLJAVANJEM

USING REINFORCEMENT LEARNING TO TRAIN AN AGENT FOR THE *Obstacle Tower* ENVIRONMENT

Predrag Njegovanović, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Učenje uslovljavanjem u današnjem poretku stvari kada je veštačka inteligencija u velikom usponu predstavlja povoljno polje za nova istraživanja.. Jedan od problema koji je rešavan poslednjih godinu ili dve jeste problem kontrole okruženja ili navigacije. U ovom radu predstavljen je jedan vid rešenja problema navigacije i generalizacije u trodimenzionalnom okruženju pri postojanju ograničenja nagrada, formiranjem autonomnog agenta tehnikama dubokog učenja. Evaluacija performansi agenta izvršena je poredbom sa ljudskim performansama i rezultatima već opisanih u propratnim naučnim radovima.*

Ključne reči: *Učenje uslovaljavanjem, trodimenzionalno okruženje, kontrola okruženja, navigacija, ograničene nagrade, neuronske mreže*

Abstract – *Reinforcement learning in today's order of things when artificial intelligence is on the rise is a favorable field for new research. One of the problems that was trying to be solved in the last year or two is the problem of environmental control or navigation. This paper presents one form of solution to the problem of navigation and generalization in a three-dimensional environment is presented while there are limits to rewards, by forming an autonomous agent with deep learning techniques. An evaluation of the agent's performance was performed by comparing it with the human performance and results already described in the accompanying scientific papers*.

Keywords: *Reinforcement learning, three-dimensional environment, environment control, navigation, sparse rewards, neural network*

1. UVOD

Učenje uslovljavanjem je računarski način učenja iz okruženja sa fokusom na postizanje zadatog cilja. S tim na umu učenje uslovljavanjem svoje primene pronalazi u zadacima optimalnog upravljanja, optimizovanja i monitoringa. Problemi učenja uslovljavanjem počinju sa potrebom za još većom količinom podataka, koja uglavnom može biti dobijena kroz simulacije, i iz tog razloga povezana je sa oblastima koja mogu pružiti ovakva okruženja, npr. video igre i robotika.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Aleksandar Kovačević, vanr. prof.

Pored ogromne količine podataka reprodukcija rezultata iz naučnih radova, nestabilnost modela kao i interpretacija i detekcija grešaka jesu problemi sa kojima se ova oblast mašinskog učenja susreće. Sve ovo jesu razlozi zbog kojih učenje uslovljavanjem nije još pronašlo svoje mesto u industriji i nije isplativo u velikoj meri [1]. Kako su spomenuta okruženja za simulaciju realnog okruženja, poslednja istraživanja svoj fokus stavljaju na posebno ponašanje u uslovima koji oponašaju realan prostor i snalaženje u istom. Naučiti računarske sisteme kako da nauče važne koncepte u svojoj interakciji sa okruženjem predstavlja dodatan problem koji je i dalje u fokusu istraživanja. Pored, ali i zbog svih ovih problema učenje uslovljavanjem predstavlja najuzbudljiviju oblast mašinskog učenja.
Tema ovog rada dotiče se svih nabrojanih problema i stavlja akcenat na učenje kretanja softverskog agenta u trodimenzionalnom prostoru pri realnom osvetljenju i znatnom većem nivou generalizacije prostora.

2. POSTOJEĆA REŠENJA

Korišćenje tehnika dubokog učenja u učenju uslovljavanjem dovelo je do razvoja autonomnih agenata čije su performanse u igrama, pre svega Atari igrama, prestigle ljudske. Sve Atari igre su dvodimenzionalne prirode i njihova stanja su potpuno vidljiva agentu u svakom trenutku (engl. *fully observable*), što podrazumeva da agentu nije potrebna memorija o prethodnim stanjima kako bi doneo optimalnu odluku.

Za razliku od ovakvih okruženja, pravi svet je trodimenzionalan i donošenje optimalne odluke zahteva pozivanje na niz od prethodnih *n* stanja. Ovakva okruženja su razmatrana u narednom radu, koji su u potpunosti ili jednim delom dotakao problema navigacije autonomnog agenta kroz prostor.

Rad koji se bavio tematikom primene pseudo nagrada na kretanje agenata jeste rad *Large-Scale Study of Curiosity-Driven Learning* od strane Burda at el. [2] sa kraja 2018. godine. Autori rada su predstavili opširnu studiju u kojoj su obuhvatili veliki broj okruženja, kako 2D tako i 3D, testirajući hipotezu da česte spoljašnje nagrade koje agent dobija od okruženja nisu skalabilne već da je neophodno agentu pružiti unutrašnje nagrade. Ovaj tip nagrada su oni nazvali znatiželja agenta (engl. *curiosity*). Znatiželja agenta se može interpretirati u kontekstu problema o kom pričamo u ovom radu i kao želja za posetom novih stanja i istraživanja okruženja. Autori definišu zanitželju kao unutrašnju nagradu koju definiše sam agent, koja

predstavlja predviđanje posledica akcija u zavisnosti od trenutnog stanja. Kreiranje unutrašnjih nagrada zadatak je modula koji je nazvan *Intrinsic-Curiosity module* (ICM). Model nad kojim se vrše eksperimenti je u ovom radu u drugom planu. Fokus rada je deo modela koji vrši ekstrakciju *feature*-a iz vizuelnih signala. Rezultati opsežnog eksperimenta pokazali su da agenti uspevaju da ostvare optimalnu politiku čak i ukoliko koriste samo generisane unutrašnje nagrade. Korišćenje samo unutrašnjih nagrada pokazalo se korisnim u Atari okruženjima, Super Mario okruženju i dr. Naravno, moguće je koristiti spoljašnju i unutrašnju nagradu u kombinaciji, što je poželjno u nekim okruženjima koja su dizajnirana sa namerom eksploatacije spoljašnje nagrade.

3. METODOLOGIJA

U ovoj sekciji pristupa se detaljnom opisivanju okruženja sa kojim agent vrši interakciju i ideja koje se kriju iza implementacije ovakvog zadatka učenja uslovljavanjem. Agent je implementiran u programskom jeziku *Python* [3]. Programski jezik *Python* postao je defakto *go-to* jezik za razvoj modula mašinskog učenja i aplikacija veštačke inteligencije zbog brzine razvijanja prototipa rešenja do postojanja mnogobrojnih biblioteka koje podržavaju razvoj.

3.1. Obstacle Tower video igra

Video okruženje *Obstacle Tower* [4] je trodimenzionalna video igra gde korisnik kontroliše agenta sa pogledom kamere iz trećeg lica (slika 1). Okruženje se sastoji od 100 spratova (prve verzije igre su sadržale samo 24 sprata), pri čemu agent počinje sa nultog sprata.

Slika 1. Igra Obstacle Tower

Cilj igre je rešiti toranj, odnosno proći kroz svih 100 spratova. Svaki sprat sadrži makar sobu u kojoj počinje sprat i u kojoj se završava. Spratovi mogu sadržati više soba od kojih neke mogu sadržati slagalice, neprijatelje, prepreke ili mogu zahtevati ključ kako bi se nastavilo dalje. Sa porastom spratova kompleksnost igre raste. Kretanje agenta u okruženju ograničeno je *in-game* vremenom koje se nadoknađuje rešavanjem sprata ili skupljanjem loptica (engl. *time orbs*) koje su nasumično razbacane po sobama.

Okruženje se renderuje koristeći osvetljenje i senčenje u realnom vremenu sa mnogo detaljnijim teksturama i modelom. Okruženje *Obstacle Tower* uključuje u sebe nekoliko varijacija koje se ogledaju kroz teksture, svetlosne uslove i geomtriju objekata.

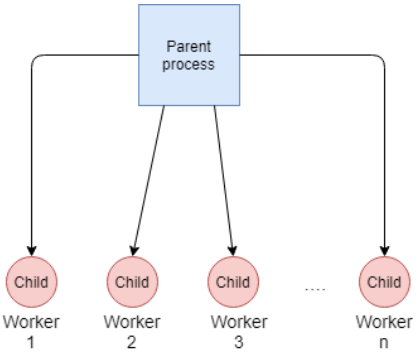
Prostor stanja agenta se sastoji od dva tipa informacija. Prvi tip observacije jeste renderovana piksel reprezentacije slike okruženja iz perspektive trećeg lica. Slika je renderovana u veličini 168x168 kao RGB slika (sadrži sva tri kanala boja, crveni, zeleni i plavi). Pored slike okruženja, drugi tip podataka jeste vektor pomoćnih vrednosti. U pomoćne informacije spadaju: broj ključeva koje agent poseduje u datom trenutku, vreme koje mu je preostalo u okviru trenutne igre i trenutni sprat na kom se nalazi.

Prostor akcija agenta je multi-diskretan što znači da se sastoji od skupa manjih diskretnih prostora akcija čija unija odgovara jednoj akciji u okruženju. Podprostori akcija obuhvataju sledeće skupove akcija: kretanje unapred, kretanje unazad, bez kretanja, zatim kretanje ulevo, kretanje udesno, mirovanje, skok ili bez skoka i rotacija kamere u smeru kazaljke na satu, obrnuto od kazaljke sata i bez rotacije kamere.

Okruženje *Obstacle Tower* jeste okruženje sa "retkim nagradama". *Obstacle Tower* pruža agentu nagradu +1 ukoliko uspešno završi ceo sprat, a +0.1 ukoliko uspešno otvori vrata, reši slagalicu ili pokupi ključ. Razređenost nagrada stavlja dodatan akcenat na korišćenje unutrašnjih nagrada i posebno oblikovanje nagrade dobijene od strane okruženja.

3.2. Paralelizacija okruženja

Pri rešavanju zadataka učenja uslovljavanjem potrebno je kontrolisati okruženje. Pod kontrolom okruženja podrazumeva se izdavanje komandi za pokretanje sledećeg koraka izvršavanja, postavljanje okruženja u početno stanje i uzorkovanje akcija iz prostora akcija. Na slici 2. predstavljen je na konceptualnom nivou dijagram koji opisuje rad modula za paralelizaciju.



Slika 2. Dijagram komunikacije između procesa

Svaki *worker* otvara po jednu instancu *Obstacle Tower* okruženja. *Parent* proces održava kontrolu nad *child* procesima i šalje im komande koje je potrebno izvršiti. Komande koje *parent* proces šalje su komande kontrole nad okruženjem. U ove komande spadaju komanda uzorkovanja koja vraća nasumičnu akciju iz prostora akcija, zatim komanda narednog koraka koja govori okruženju da pređe u naredni vremenski korak vraćajući pri tom observacije i komanda resetovanja koja postavlja okruženje u početno stanje. Prilikom slanja komandi, *parent* proces prolazi kroz listu svih *child* procesa i šalje im signal. Dobijeni podaci se agregiraju u *parent* procesu u obliku tenzora i prosleđuju dalje.

3.3. Model i interfejs ka modelu

Modeli su definisani kroz *PyTorch*, biblioteku za mašinsko učenje. Konvolutivna neuronska mreža ima strukturu zasnovanu na radu [5] objavljenom u *Nature* časopisu. Prvi sloj neuronske mreže ima 32 filtera sa veličinom kernela 8x8 i korakom pomeranja kernela 4. Drugi sloj mreže ima 64 filtera, kernel 4x4, korak 2 i treći sloj takođe ponavlja 64 filtera sa veličinom kernela 3x3 i korakom 1. Na izlaz iz svakog kernela se primenjuje *LeakyReLU* aktivaciona funkcija.

Između slojeva je upotrebljena i *batch normalization* tehnika [6] za uklanjanje pomeraja u distribuciji podataka. Rekurentna neuronska mreža inicijalizovana je sa GRU ćelijama i veličinom matrice za stanje ćelije od 512 kolona. Zadržan je samo jedan sloja rekurentne mreže. Linearni slojevi za aproksimaciju funkcije stanja i politike su definisani shodno traženim dimenzijama tj. bile su uslovljenje veličinom izlata rekurentne neuronske mreže.

Arhitektura ICM modela ispraćena je kroz naučni rad koji ju je i predstavio. Blok za ekstrakciju *feature*-a je identična konvolutivna mreža koju koristi i osnovni model uz razliku u izlaznoj transformaciji, gde izlazni vektor sadrži 288 elemenata. Forward model je kombinacija tri potpuno povezana sloja sa 256, 512 i 288 neurona. Aktivacija svakog sloja je *LeakyReLU*. Inverse model jeste potpuno povezana mreža sa četiri sloja izlaznih dimenzija 256, 512, 512 i veličinom prostora akcija, respektivno. Izlazna aktivaciona funkcija jeste *softmax* funkcija.

Interfejs ka modelima sakriva modele i otkriva metode za manipulaciju istima. Metode koje Tower Agent pruža su: metoda za propagaciju podataka unapred kroz osnovni model, metoda za propagaciju podataka unapred kroz ICM model, metoda za računanje funkcije gubitka sinhronog *actor-critic* algoritma i metoda za računanje funkcije gubitka *proximal policy optimizazion* algoritma.

3.4. Skladištenje observacija

Skladišenje observacija poslužilo je kao prikladan način za uzorkovanje trajektorija jednog od više paralelnih okruženja kao i za računanje očekivanih nagrada za svaki vremenski korak do kraja epizode.

Skladište se sledeće informacije: stanje okruženja, nagrade, terminalna stanja, prediktovane vrednosti funkcije stanja, odabrane akcije za svaki vremenski korak i vrednosti politike za svaki vremenski korak. Dodatno se čuvaju i vektori stanja koji se koriste prilikom računanja funkcije greške ICM modela. Osnovne operacije su dodavanje, brisanje ali i uzorkovanje prilikom treninga agenta. Uzorkovanje iskustava se vrši za sinhronu *actor-critic* metodu kao i za *proximal policy optimization.*

3.5. Obučavanje agenta

Parametri koji kontrolišu tok obučavanja agenta su: ukupan broj vremenskih koraka agenta, količina prikupljenog iskustva, broj epoha ažuriranja, stopa učenja i algoritam obučavanja. Preostali parametri inicijalizuju ostale neophodne module aplikacije. Ukupan broj koraka agenta se uobičajeno zadaje pre početka treninga i direktno određuje dužinu treninga. Usled hardverskih ograničenja, izabrani broj koraka je 5 miliona. Uobičajeno se agent trenira sa 20-50 miliona koraka. Trening agenta započinje izračunavanjem broj ažuriranja, a dva najvažnija koraka treniranja agenta jesu prikupljanje iskustva agenta i primena iskustva na ažuriranje modela. Petlja za prikupljanje iskustva agenta iznosi 128 koraka. Zbog jednostavnosti problem je definisan kao epizodni, što podrazumeva da se prikupljanje iskustva završava kada se dođe do terminalnog stanja ili nakon ispunjenja svih 128 koraka. Biranje sledeće akcije se vrši *on-policy* pristupom, iz trenutne politike se uzorkuje nasumična akciju i primeni se na okruženje. Observacije koje okruženje pruža puštaju se kroz osnovni model i ICM model gde se kao izlaz dobijaju vrednosti funkcije stanja, politike i unutrašnje nagrade. Treba napomenuti da se pored stanja, osnovnom modelu pruža i poslednje stanje ćelije rekurentne mreže.

Na taj način održavamo zavisnost između vremenskih koraka i modelujemo sekvencu događaja u rekurentnoj neuronskoj mreži. Ovi podaci smeštaju se u bafer iskustva. Nakon prikupljanja iskustva pristupa se ažuriranju modela. Ažuriranje modela zavisi od primenjenog algoritma. U slučaju *Synchronous actor-critic* ne postoje ciklusi ažuriranja niti višesutruka ažuriranja jer priroda algoritma to ne dozvoljava. Uzorkovanjem bafera iskustva se dobijaju releventni podaci neophodni za ažuriranje modela. Definisani algoritam za optimizaciju funkcije gubitka, u slučaju oba algoritma, jeste Adam [7] i inicijalno postavljena vrednost stope učenja iznosi 0.0004 što se pokazalo kao dovoljno velika vrednost da agent počne brzo da uči ali ne i prevelika da učenje divergira. Vrednost funkcije gubitka se normalizuje brojem epoha ažuriranja u slučaju PPO algoritma i vrši se propagacija unazad. Svi delovi funkcije gubitka se čuvaju u definisanim tenzorima radi prikazivanja na grafikonima.

4. EVALUACIJA REŠENJA I DISKUSIJA REZULTATA

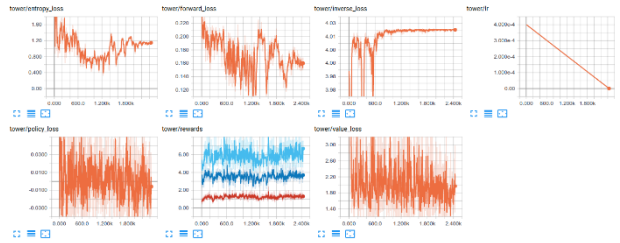
Standardna procedura po završetku obučavanja modela jeste provera performansi modela koja će pokazati da li model zadovoljava postavljene kriterijume.

Prosečne ljudske performanse u okruženju *Obstacle Tower* prezentovane su u već spomenutom naučnom radu koji je pratio objavu okruženja [4] i iznose 15 spratova po prosečnoj igračkoj sesiji bez dinamičkog generisanja nivoa. Sa istim postavkama okruženja najbolji modeli zasnovani na čistom *PPO* i *Rainbow* algoritmu dostižu 5 odnosno 7 spratova. Model *Tower agent* je obučavan u perspektivama sa pogledom iz trećeg lica i sa pogledom iz prvog lica. Pretpostavka vezana za perspektivu iz prvog lica jeste da će otkloniti suvišan prikaz prostora a s tim i *feature*-e koje pruža perspektiva iz trećeg lica i da će time ostvariti bolje performanse. Modeli su puštani 10 puta i uzimala se maksimalna vrednost broja spratova koje je agent uspeo da reši. Rezultati su prikazani tabelom 1.

	Proximal Policy Optimization	*Synchronous actor-critic*
First person perspective	1	1
Third person perspective	3	2

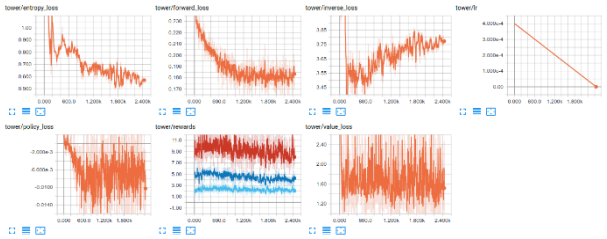
Tabela 1. Prosečan broj rešenih spratova u zavisnosti od perspektive kamere

Na osnovu rezultata iz table 1. izvlače se zaključci da je postavljena hipoteza zadovoljena u slučaju kada je pogled na okruženje u perspektivi trećeg lica uz bolje rezultate *PPO* algoritma. Pokazuje se da agent ostvaruje lošije performanse u perspektivi prvog lica. Posmatranjem agenta u fazi zaključivanja primećuju se problemi koje ima sa samim okruženjem. Često tokom kretanja, realistično osvetljenje i probijanje ivica okruženja "hvata" nepotrebne *feature*-e koji u krajnjem slučaju ne vode do dobrih rezultata. Grafici (slika 3) koji opisuju obučavanje *A2C* algoritmom pokazuju nestabilnost prilikom treninga i veliko oscilovanje greške funkcije politike.



Slika 3. Tok obučavanja A2C algoritma za perspektivu iz prvog lica

Performanse agenta obučavanim *PPO* algoritmom sa perspektivom kamere iz trećeg lica, što i jeste podrazumevano podešavanje, zadovoljava postavljenu hipotezu o prelasku jednog sprata ali ne uspeva da ostvari rezultate prezentovane u naučnom radu koji prati okruženje. Maksimalan sprat koji agent dostiže jeste sprat broj tri. Međutim uzevši u obzir da je trening agenta koji se opisuje u ovom radu trajao deset puta kraće, rezultati su zadovoljavajući. Proces obučavanja kroz grafike funkcija gubitaka i nagrade prikazan je slikom 4.



Slika 4. Tok obučavanja PPO algoritam za perspektivu iz trećeg lica

5. ZAKLJUČAK

Kroz ovaj rad predstavljena je aplikativna primena učenja uslovljavanjem na jednom od poslednjih istraživačkih problema. Za potpunu kontrolu nad okruženjem potrebna su neka od kompleksnijih rešenja kao npr. *behavioral cloning* algoritam gde agent uči da oponaša ljudski performans u okruženju, korišćenje KL-divergence faktora umesto funkcije entropije, GAIL (engl. *Generative Adversarial Imitation Learning*) metode ili kombinacija istih. Sam proces obučavanja agenta je složen i zahtevan sa strane vremenskih i hardverskih resursa. Novi trend obučavanja agenta kroz skupljanje iskustva kroz više paralelnih okruženja zadovoljava potrebu za velikim količinama podataka ali povećava hardverske zahteve, pre svega u vidu procesorske moći, radne memorije ali najviše video memorije. Naravno da bi svi resursi bili optimalno iskorišćeni, celokupnu aplikaciju je potrebno napisati na pametan način. *Obstacle Tower* okruženje se pokazalo kao vrlo izazovno za savladavanje i spremno je da gurne granice istraživanja u učenju uslovljavanjem. Izazovi sa vizuelnog aspekta, sa aspekta kontrole i planiranja ali i nagrada koje pruža okruženje zahtevaju ozbiljno poznavanje materije i ostavljaju prostor za dalja istraživanja. Ovo okruženje svakako potkrepljuje tvrdnju da vremenom nastaju sve kompleksnija okruženja koja bivaju gurnuta na pozornicu u ovoj oblasti. Poslednji pravci istraživanja u trenutnoj postavci stvari obuhvataju probleme vizije, planiranja, kontrolisanja trodimenzionalnih okruženja kao i okruženja sa retkim nagradama. Pored svih nabrojanih problema, implementacionih i aplikativnih, učenje uslovljavanjem jeste jedna od najuzbudljivijih oblasti veštačke inteligencije i veliki uspesi na ovom polju se tek očekuju.

6. LITERATURA.

[1] AY Ng. (2017, December 14). Practical applications of reinforcement learning in industry. Retrieved June 24, 2019 from https://www.oreilly.com/ideas/practical-applications-of-reinforcement-learning-in-industry

[2] Burda, Y., Edwards, H., Pathak, D., Storkey, A., Darrell, T., & Efros, A. A. (2018). Large-scale study of curiosity-driven learning. arXiv preprint arXiv:1808.04355.

[3] Python. (n.d). Retrieved from https://www.python.org/

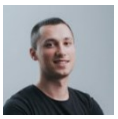
[4] Juliani, A., Khalifa, A., Berges, V. P., Harper, J., Henry, H., Crespi, A., ... & Lange, D. (2019). Obstacle Tower: A Generalization Challenge in Vision, Control, and Planning. arXiv preprint arXiv:1902.01378.

[5] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

[6] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

[7] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kratka biografija:



Predrag Njegovanović je rođen 4.10.1994. u Vukovaru, Republika Hrvatska. Osnovnu školu „Slobodan Bajić Paja" završio je 2009. godine. Iste godine upisuje srednju tehničku školu „Nikola Tesla", smer elektrotehničar računara. Srednju školu završava 2013. godine i upisuje osnovne akademske studije na smeru Računarstvo i automatika, Fakulteta tehničkih nauka u Novom Sadu. Zvanje diplomirani inženjer elektrotehnike i računarstva stiče 2017. godine, sa prosečnom ocenom 9,59, uz specijalizaciju računarske nauke i informatika, modul inteligentni sistemi. Nakon toga, iste godine, upisuje master akademske studije na Fakultetu tehničkih nauka u Novom Sadu, odsek Elektrotehnika i računarstvo, smer Računarstvo i automatika, modul Inteligentni sistemi. Položio je sve ispite predviđene planom i programom master studija uz prosečnu ocenu 10,0.