Podržano učenje: Alphastar

Balen, Ivan

Sveučilište Jurja Dobrile u Puli

Fakultet informatike u Puli

Ivan Balen

Podržano učenje: Alphastar

Završni rad

Pula, rujan, 2019. godine

Sveučilište Jurja Dobrile u Puli
Fakultet informatike u Puli

Ivan Balen

Podržano učenje: Alphastar

Završni rad

JMBAG: 0303054222, vanredni student

Studijski smjer: Informatika

Predmet: Modeliranje i simulacija

Znanstveno područje: Društvene znanosti
Znanstveno polje: Informacijske i komunikacijske znanosti
Znanstvena grana: Informacijski sustavi i informatologija
Mentor: doc. dr. sc. Darko Etinger
Komentor: dr. sc. Nikola Tanković

Pula, rujan, 2019. godine

IZJAVA O AKADEMSKOJ ČESTITOSTI

Ja, dolje potpisani Ivan Balen, kandidat za prvostupnika informatike ovime izjavljujem da je ovaj Završni rad rezultat isključivo mojega vlastitog rada, da se temelji na mojim istraživanjima te da se oslanja na objavljenu literaturu kao što to pokazuju korištene bilješke i bibliografija. Izjavljujem da niti jedan dio Završnog rada nije napisan na nedozvoljen način, odnosno da je prepisan iz kojega necitiranog rada, te da ikoji dio rada krši bilo čija autorska prava. Izjavljujem, također, da nijedan dio rada nije iskorišten za koji drugi rad pri bilo kojoj drugoj visokoškolskoj, znanstvenoj ili radnoj ustanovi.

Student

U Puli, rujan, 2019. godine

_____

IZJAVA
o korištenju autorskog djela

Ja, Ivan Balen dajem odobrenje Sveučilištu Jurja Dobrile u Puli, kao nositelju prava iskorištavanja, da moj završni rad pod nazivom „Machine Reinforcement“ koristi na način da gore navedeno autorsko djelo, kao cjeloviti tekst trajno objavi u javnoj internetskoj bazi Sveučilišne knjižnice Sveučilišta Jurja Dobrile u Puli te kopira u javnu internetsku bazu završnih radova Nacionalne i sveučilišne knjižnice (stavljanje na raspolaganje javnosti), sve u skladu s Zakonom o autorskom pravu i drugim srodnim pravima i dobrom akademskom praksom, a radi promicanja otvorenoga, slobodnoga pristupa znanstvenim informacijama.
Za korištenje autorskog djela na gore navedeni način ne potražujem naknadu.

U Puli, rujan, 2019. godine

Potpis

_____

Sadržaj

1 Uvod

Machine reinforcement learning ili strojno podržano učenje je područje strojnog učenja koje se bavi proučavanjem načina na koji softverski agenti trebaju poduzimati akcije u okruženju kako bi maksimizirali neki pojam kumulativne nagrade. To je jedna od tri osnovne paradigme strojnog učenja, osim učenja pod nadzorom i učenja bez nadzora. Učenje pod nadzorom razlikuje se po tome što označeni parovi ulaza/izlaza ne moraju biti predstavljeni, a sub-optimalne radnje ne moraju biti izričito ispravljene. Umjesto toga, fokus je pronalaženje ravnoteže između istraživanja (neizgrađenog teritorija) i eksploatacije (postojećeg znanja). Okruženje se obično formulira kao Markov postupak odlučivanja, jer mnogi algoritmi učenja za podržanje u ovom kontekstu koriste dinamičke tehnike programiranja. Glavna razlika između klasičnih metoda dinamičkog programiranja i algoritama učenja podržanja je u tome što oni ne pretpostavljaju točan matematički model Markovog postupka odlučivanja i ciljaju na odjele gdje točne metode postaju neizvodljive. Iako su neuronske mreže odgovorne za nedavne pomake u problemima poput računalnog vida, strojnog prevođenja i predviđanja vremenskih serija, one se mogu kombinirati i s algoritmima učenja za podržanje kako bi se stvorilo nešto zapanjujuće poput AlphaGo, OpenAI, Alphastar itd. Podržanje učenja odnosi se na algoritme usmjerene na ciljeve, koji uče kako postići složen cilj ili maksimizirati cilj uz određenu dimenziju kroz više koraka; na primjer, maksimizirajte osvojene bodove u igri tijekom određenog broja poteza. Oni mogu započeti u praznoj ploči i pod pravim uvjetima postižu nadljudske performanse. Poput djeteta koje potiče pljuvanje i slatkiše, i ovi algoritmi su kažnjeni kada donose pogrešne odluke i nagrađeni su kada donose ispravne odluke, to u principu definira zadaću podržanja. Podržani algoritmi koji uključuju duboko učenje mogu pobijediti svjetske prvake u igrama, kao i ljudske stručnjake u drugim igrama koje se ne igraju preko osobnog računala. Iako to može zvučati trivijalno, njihovo poboljšanje u odnosu na njihova prethodna dostignuća nije lagan pomak, kao i njihovo stanje tehnike koje brzo napreduje. Algoritmi podržanja također rješavaju težak problem povezanosti neposrednih radnji s odgođenim povratima. Poput ljudi, i algoritmi za učenje podržanja ponekad moraju pričekati neko vrijeme da bi ugledali plod svojih odluka. Djeluju

u okruženju s odgođenim povratkom gdje može biti teško shvatiti koja akcija vodi do pobjedničkog ishoda tijekom mnogih vremenskih koraka. Može se očekivati da algoritmi za učenje podržanja djeluju bolje i bolje u više dvosmislenim stvarnim okruženjima, istovremeno birajući iz proizvoljnog broja sve moguće radnje, a ne iz ograničenih mogućnosti videoigre, točnije s vremenom očekujemo da će oni biti vrijedni za postizanje ciljeva u stvarnom svijetu. Skymind primjenjuje učenje dubokog podržanja na simulacijama slučajeva u stvarnom svijetu kako bi pomogao tvrtkama da optimiziraju način izgradnje tvornica, zaposlenih telefonskih centara, postavljanja skladišta i lanaca opskrbe i upravljanjem prometnih tokova. Osim primjene u video igrama, strojno podržanje se također primjenjuje u područjima poput teorije upravljanja, operativnog istraživanja, teorije informacija, optimizacije temeljene na simulaciji, multiagencijskih sustava, statistike i genetskih algoritama. Područje ekonomije i video igara može upotrijebiti strojno podržanje da bi se objasnilo kako može nastati ravnoteža pod ograničenom racionalnošću između agenata. Pravila koja se moraju poštivati su stohastička, a promatranje obično uključuje skalarnu, neposrednu nagradu povezanu s posljednjim prijelazom. U mnogim slučajevima pretpostavlja se da agent promatra trenutno stanje okoliša, a ako ga ne promatra, agent ima djelomično promatranje. Ponekad je skup radnji koji je dostupan agentu ograničen, stoga se nulta ravnoteža ne može smanjiti (npr. ako je trenutna vrijednost agenta 3, a prijelaz stanja smanjuje vrijednost za 4, prijelaz neće biti dopušten).

Agent za učenje djeluje s okolinom u diskretnim vremenskim koracima. U svakom trenutku t, agent prima opažanje, što obično uključuje nagradu. Zatim odabire akciju iz skupa dostupnih radnji koja se nakon toga šalje u okruženje. Okolina se prebacuje u novo stanje,a nagrada je povezana s prijelazom. Cilj sredstva za učenje podržanja je prikupiti što više nagrada. Agent može odabrati bilo koju radnju kao funkciju za analizu povijesti svojih radnji. Kad se učinak agenta usporedi s onim koji ima agent koji djeluje optimalno, razlika u učinku rađa pojam žaljenja. Da bi djelovao u blizini optimalno, agent mora razmišljati o dugoročnim posljedicama svojih postupaka(maksimizirati budući prihod), iako neposredna nagrada povezana s tim događajem može biti negativna. Stoga je strojno podržanje posebno prikladno za probleme koji uključuju dugoročni kompromis nasuprot kratkoročnom kompromisu nagrađivanja.

2 Definicije strojnog podržanja učenja

Učenje podržanja se može razumjeti upotrebom koncepta agenta, okruženja, stanja, radnji i nagrade, što će sve biti objašnjeno u nastavku. Agent poduzima akcije; na primjer, dron bez pilota koji isporučuje pošiljku ili Super Mario koji se kreće u video igri, algoritam je agent. U stvarnom životu smo svi agenti i pokušavamo doći do nagrade, tako se ponaša i agent u strojnom podržanju. Akcija je skup svih mogućih poteza koje agent može izvršiti. Radnja je gotovo nerazumljiva, ali treba napomenuti da agenti biraju svoje akcije preko popisa svih mogućih radnji. U video igrama popis može uključivati trčanje u desno ili lijevo, skakanje u vis ili nisko, naginjanje ili stajanje. Popis na burzama može uključivati kupnju, prodaju ili držanje bilo kojeg od vrijednosnih papira i njihovih derivata. Pri rukovanju zračnim letjelicama bez pilota, alternative bi uključivale mnogo različitih brzina i ubrzanja u 3D prostoru. Faktor popusta je faktor koji se množi s budućim nagradama kako ih agent postepeno otkriva i kako bi se prigušio učinak nagrade na izbor aktivnosti agenta, zatim se postavlja pitanje zašto on tako funkcionira, a odgovor na to pitanje je da je on osmišljen i implementiran tako da buduće nagrade vrijede manje od neposrednih nagrada; tj.provodi se svojevrsni kratkoročni hedonizam u agentu. Često se izražava grčkim gama slovima : γ.

Ako je γ 0,8, a nagrada dobiva 10 bodova nakon 3 vremenska koraka, sadašnja vrijednost te nagrade je $0,8^3$ x 10. Faktor popusta od 1 učinio bi buduće nagrade vrijedne isto toliko koliko i neposredne nagrade. Ovdje se borimo protiv zakašnjenja zadovoljstva. Idući koncept koji utječe na strojno podržanje je koncept okoliša, a to bi bio nekakav svijet kroz koji se agent kreće. Okolina uzima trenutno stanje i radnje agenta kao ulaz i vraća kao rezultat nagradu agenta i njegovo sljedeće stanje. Ako ste agent, okolina bi mogla biti zakon fizike i pravila društva koja obrađuje vaše postupke i utvrđuje njihove posljedice. Stanje je konkretna i neposredna situacija u kojoj se agent nalazi; tj.određeno mjesto i trenutak, trenutna konfiguracija koja stavlja agenta u odnos na druge značajne stvari kao što su alati, prepreke, neprijatelji ili nagrade,to može biti bili koja trenutna situacija koja vraća okoliš ili bilo koja buduća situacija koja nam govori je li agent ikada bio na krivom mjestu u krivo vrijeme.

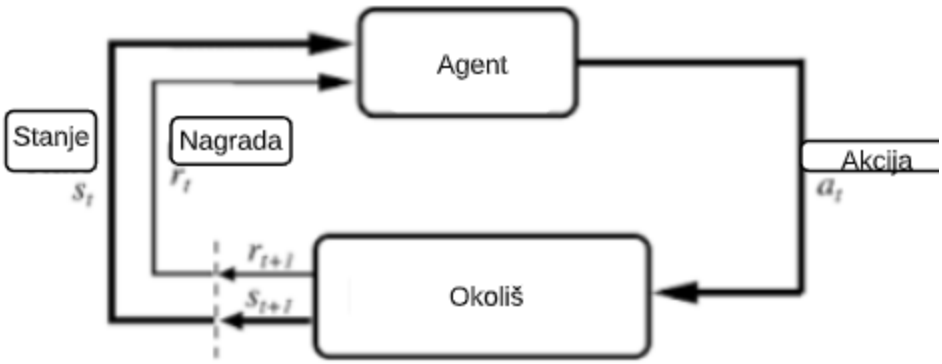
Koncept nagrade je povratna informacija kojom mjerimo uspjeh ili neuspjeh radnji agenta. Na primjer, u video igri, kada Mario dodirne novčić, osvaja bodove. Iz bilo kojeg stanja, agent šalje izlaz u obliku radnji u okolinu, a okruženje vraća agentovo novo stanje(koje je nastalo djelovanjem na prethodno stanje) kao i nagrade, ako ih ima. Nagrade mogu biti momentalne ili odgođene, i one učinkovito ocjenjuju djelovanje agenta. Koncept pravila je strategija koju agent koristi za određivanje sljedeće akcije na temelju trenutnog stanja. To preslikava različita stanja u akciji, radnje koje obećavaju najveću nagradu. Vrijednost je očekivani dugoročni povrat s popustom, za razliku od kratkoročne nagrade. Definira se kao očekivani dugoročni povrat trenutnog stanja u skladu s pravilom. Nagrade snižavamo ili smanjujemo njihovu procijenjenu vrijednost, i analiziramo što se dalje događa u budućnosti. Kada pogledamo faktor popusta, dugoročno gledano po teoremu Keynesa: „Svi smo mrtvi.“ Zbog toga snižavamo buduće nagrade. Q-vrijednost ili vrijednost akcije je slična običnoj vrijednosti, osim što ima dodatni parametar, trenutnu radnju, to se odnosi na dugoročni povrat trenutnog stanja s poduzimanjem akcija pod nadzorom koncepta pravila. Dakle sve ove koncepte možemo svrstati u nekakvo okruženje, a ta okruženja su funkcije koje transformiraju akciju koja je u trenutnom stanju u sljedeće stanje, i nakon toga nagrađuju agenta. Agenti su funkcije koje novo stanje transformiraju i nagrađuju u sljedeću radnju. Možemo znati funkciju agenta, ali ne možemo znati okoliš. To je crna kutija u kojoj vidimo samo ulaze i izlaze. To je poput odnosa većine ljudi s tehnologijom : znamo što čini, ali ne znamo kako to funkcionira. Učenje podržanja predstavlja pokušaj agenta da približi funkciju okoliša, tako da mi možemo slati akcije u okruženje crne kutije koje povećavaju nagradu koju on zaslužuje.

Glavne točke u strojnom podržanju učenja :

-Ulaz : Ulaz treba biti početno stanje iz kojeg će model krenuti.
-Izlaz : Postoji mnogo mogućih rezultata jer postoje različita rješenja određenog problema.

-Obuka : Trening se temelji na ulaganju.

-Model : Model nastavlja učiti i najbolja solucija je odlučena na najvećoj nagradi.



Slika 1. Povratna veza između agenta i okoliša.

U gornjoj petlji povratne informacije pretplatnici označavaju vremenske korake t i t +1, od kojih se svaki od njih odnosi na različita stanja : stanje u trenutku t i stanje u trenutku t +1. Za razliku od drugih oblika strojnog učenja – poput učenja pod nadzorom i bez nadzora, o podržanju učenja može se razmišljati samo uzastopno u smislu parova stanja i djelovanja koji se javljaju jedan za drugim. Podržanje učenja ocjenjuje akcije prema rezultatima koje agenti daju. Usmjeren je na cilj, a njegov je cilj naučiti slijedeće radnje koje će voditi agenta da postigne svoj cilj ili maksimizira svoju ciljnu funkciju, stoga su navedeni nekoliko primjera ; u video igrama cilj je završiti igru s najviše bodova, tako da će svaki dodatni bod dobiven tijekom igre utjecati na ponašanje agenta, tj.agent može naučiti da bih trebao pucati borbene brodove, dodirnuti kovanice ili izmicati se od meteora da bi maksimizirao rezultat svoje performanse. U stvarnom svijetu, cilj bi mogao biti da robot putuje od točke A do točke B, a svaki centimetar gdje se robot približi točki B se može računati kao dodatni bod. Primjer objektivne funkcije za učenje podržanja, tj. način na koji se određuje cilj je sljedeći :

$$\sum_{t=0}^{t=\infty} \gamma^t r\big(x(t), a(t)\big)$$

Formula 2. Suma funkcije za nagrađivanje agenta

Sumiranjem nagradne funkcije r preko t, koja označava vremenske korake dobijemo funkciju koja izračunava svu nagradu koju bismo mogli dobiti provodeći npr. Igru. Ovdje je x stanje u određenom vremenskom koraku, dok je a radnja poduzeta u tom stanju. Funkcija r je funkcija nagrađivanja za x i a.

Strojno učenje podržanja razlikuje se od učenja pod nadzorom i bez nadzora po tome što tumači inpute. Možemo ilustirati njihovu razliku opisujući ono što oni uče o nečemu. Nenadzirano učenje je vrsta učenja koja govori da je jedna stvar poput nekakve druge stvari (Algoritmi uče sličnosti bez imena, a ekstenzijom mogu uočiti obrnuto i izvršiti otkrivanje anomalije prepoznavanjem onoga što je neobično ili različito). Nadzirano učenje koristi algoritme koji uče korelacije između instanci podataka i njihovih oznaka, oni zahtijevaju obilježeni skup podataka. Te se oznake upotrebljavaju za nadzor i ispravljanje algoritma, jer čini pogrešne pretpostavke prilikom predviđanja oznaka. Strojno učenje funkcionira na princip da govori agentu : „Jedite tu stvar jer je dobar okus i održavat će vas duže na životu.“ (radnje koje se temelje na kratkoročnim i dugoročnim nagradama, kao što su količina kalorije koju unosite ili duljina vremena koju poživite). Strojno podržanje učenja se može smatrati nadziranim učenjem samo u okruženju rijetkih povratnih informacija.

2.1 Vrste strojnog podržanja

Postoje 2 vrste podržanja, pozitivno i negativno.

Pozitivno – pozitivno podržanje definira se kao kad se neki događaj dogodi zbog određenog ponašanja, povećava snagu i učestalost ponašanja. Drugim riječima, pozitivno utječe na ponašanje. Prednosti učenja podržanja su : maksimizacija performanse,održavanje promjene u dužem vremenskom razdoblju, a nedostaci su preopterećenje stanja što može umanjiti rezultate.

Negativno – Negativno podržanje definira se kao jačanje ponašanja jer se negativno stanje zaustavlja ili se izbjegava. Prednosti ovog učenja su povećanje mogućnosti ponašanja agenta, pružanje otpora prema minimalnim standardima performansi. Nedostaci ovog podržanja su da se pruža samo dovoljno informacija da se ispuni minimalno ponašanje. Poznat je model okoliša, ali analitičko rješenje nije dostupno, daje

se samo simulacijski model okoliša (predmet optimizacije temeljene na simulaciji). Jedini način prikupljanja podataka o okolišu je interakcija s njim.

2.2 Algoritmi za kontrolu učenja

Pravilo : Odabir radnje agenta da se modelira kao karta koja se zove pravilo

$$\pi: Axs \rightarrow [0,1]$$
$$\pi(a,s) = p_r(a_t = a | s_t = s)$$

Formula 3. Karta pravila daje vjerojatnost poduzimanja akcije kada se poziva „a“ i kada se poziva „s“

Funkcija stanja-vrijednosti :

$$V_\pi(s) = E[R] = E\left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s\right]$$

Formula 4. Funkcija vrijednosti se definira tako što počinjem s jednim stanjem „s“ i očekivajuća povratna informacija slijedi pravilo „π“. Time funkcija vrijednosti procjenjuje koliko je ona dobra i potrebna u danom stanju.

$$R = \sum_{t=0}^{\infty} \gamma^t r_t$$

Formula 5. Slučajna varijabla „R“ označava povrat i definira se kao zbroj budućih nagrada s popustom

Algoritam mora pronaći pravilo s maksimalnim očekivanim povratom. Iz teorije MDP-a poznato je da se, bez gubitka općenitosti, pretraživanje može ograničiti na skup takozvanih stacionarnih pravila. Pravilo je nepokretno ako raspodjela akcija koja se vraća ovisi samo o posljednjem posjećenom stanju(iz povijesti promatrača). Potraga se može dalje ograničiti na determinirana stacionarna pravila. Deterministička stacionarna pravila determinirano odabiru akcije na temelju trenutnog stanja.

Brute force (iscrpna pretraga) : Pristup grube sile podrazumijeva dva koraka.

1 – Za svaki mogući pravilnik, uzorak vraća podatak dok ga slijedimo. Odabirom pravila s najvećim očekivanom povratom je jedna od mogućnosti koju možemo koristiti. Jedan od problema koji se može izdvojiti je taj što broj pravila može biti velik pa čak i beskonačan. Drugi je način da različite varijante prinosa podataka budu veće, što zahtijeva veliku količinu uzoraka da bi se precizno procijenio povrat svih pravila.

2 - Ovi se problemi mogu poboljšati ako se pretpostavi nekakva struktura i omogući uzorcima dobivenim iz jednog pravila da utječu na procjenu napravljenu za druge. Dva glavna pristupa za postizanje toga su procjena vrijednosti i izravno traženje pravila.

Funkcija vrijednosti : Pristupi funkcijskim vrijednostima pokušavaju pronaći pravilo koje maksimizira povrat održavanja skupa procjena, pritom očekivajući i povrat pravila utrošenih da se to omogući. Ove se metode oslanjaju na teoriju MDP-a gdje se optimalnost definira u smislu jačem od gornjeg. Pravilo se naziva optimalnom ako postigne najbolji očekivani povrat iz bilo kojeg početnog stanja, a optimalno pravilo se uvijek može naći među stacionarnim pravilima.

$$V^{\pi}(s) = E[R|s, \pi]$$

Formula 6. Da se formalno definira optimalnost, mora se definirati i vrijednost pravila „π“.

$$V(s) = \max_{\pi} V^{\pi}(s)$$

Formula 7. „R“ označava povratak povezan sa slijeđenjem „π“ iz početnog stanja „s“, dok „V*“ se definira kao najveća moguća vrijednosti

Pod pretpostavkom potpunog poznavanja MDP-a, dva osnovna pristupa izračunavanju optimalne akcije vrijednosti funkcija su iteracija vrijednosti i iteracija pravila. Oba algoritma računaju niz funkcija koje se konvergiraju.

2.3 Metode

Monte Carlo metoda : Monte Carlo metode mogu se koristiti u algoritmu koji oponaša iteraciju pravila. Ponavljanje pravila sastoji se od dva koraka : evaluacija pravila i poboljšanje pravila. Monte Carlo koristi se u koraku evaluacije pravila. U ovom koraku, s obzirom na stacionarno, determinističko pravilo „π“ cilj je izračunati vrijednosti funkcija za sve parove stanja akcije, pod pretpostavkom da je MDP konačan, da je na raspolaganju dovoljno memorije za smještaj vrijednosti akcije i da je problem epizodan, a nakon svake epizode novi počinje iz nekog slučajnog početnog stanja. Zatim procjena vrijednosti određenog para akcije-stanja se može izračunati prosjekom uzrokovanih povrata. Obzirom na dovoljno vremena, ovaj postupak može konstruirati preciznu procjenu funkcije akcije-vrijednosti. Ovime se završava opis koraka vrednovanja pravila. U koraku poboljšanja pravila, sljedeće pravilo dobiva se računanjem pohlepnog pravila s obzirom na stanje. Novo pravilo vraća radnju koja maksimizira određenu vrijednost. Lijena procjena u praksi se može odgoditi izračunavanjem maksimizirajućih radnji dok god se ne naiđe na nekakvu radnju koja nam je potrebna. Problemi s tim postupkom uključuju da procedura može potrošiti previše vremena na ocjenu suboptimalnog pravila. Učinkovito koristi uzorke tako da dugačka putanja poboljšava procjenu pojedinog para stanja djelovanja koji je pokrenuo putanju. Kada se događaju velika odstupanja duž putanju, konvergencija je spora. Djeluje samo u epizodnim problemima, a radi samo u malim, konačnim MDP-ovima.

Metode vremenske razlike : Prvi se problem ispravlja dopuštajući postupak promjene pravila (u nekim ili svim stanjima) prije nego što se vrijednosti postignu, ovo bi moglo biti problematično jer može spriječiti konvergenciju. Većina trenutnih algoritama to radi, stvarajući klasu generaliziranih algoritama iteracije pravila. Drugi se problem može ispraviti dopuštanjem putanja da daju svoj doprinos bilo kojem paru akcija stanja. To može u određenoj mjeri pomoći i trećem problemu, mada je bolje rješenje kada prinosi imaju veliku varijancu Suttonove vremenske razlike, to su metode koje se temelje na rekurzivnoj Bellmanovoj jednadžbi (matematička optimizacija metode poznata kao

dinamičko programiranje). Računanje metoda vremenskih razlika može biti inkrementalno (kada se nakon svakog prijelaza memorija promijeni i prijelaz se odbaci).

Osim inkrementalnog računanja imamo i batch (kada se prijelazi skupa i procjene izračunavaju jednom na temelju serije). Skupne metode, kao što je metoda vremenskih razlika najmanjeg kvadrata, mogu bolje koristiti informacije u uzorcima, dok su inkrementalne metode jedini izbor kada su batch metode neizvodljive zbog velike složenosti računanja ili memorije. Neke metode pokušavaju kombinirati dva pristupa.

$$Q(s,a) = \sum_{i=1}^{d} \theta_i \phi_i(s,a)$$

Formula 8. Koriste se metode aproksimacije funkcija. Linearno približavanje funkcije započinje s preslikavanjem dimenzionalno-ograničenog vektora svakom paru stanja akcije. Zatim, vrijednosti akcije para „s,a“ dobivaju se linearnim kombiniranjem komponenti vektora „s,a“.

Algoritmi tada prilagođavaju težine, umjesto da prilagođavaju vrijednosti povezane s pojedinačnim parovima stanja i akcije. Istraživane su metode zasnovane na idejama iz neparametrijske statistike (testovi koji ne moraju podrazumijevati normalnost distribucije podataka kojima raspolažemo). Vrijednost iteracije može se koristiti i kao polazište, što dovodi do algoritma učenja Q-a i njegovih mnogih inačica. Problem s korištenjem vrijednosti akcije je što će im trebati vrlo precizne procjene konkurentskih vrijednosti djelovanja, koje je teško dobiti kada su povratci bučni. Iako se ovaj problem donekle ublažava metodama vremenske razlike, korištenje takozvane kompatibilne funkcije metode aproksimacije, ona ugrožava općenitost i učinkovitost.

Izravno pretraživanje pravila : Alternativna metoda je izravno pretražiti nekakav podskup prostora pravila, u tom slučaju problem postaje slučaj stohastičke optimizacije. Dva dostupna pristupa su metode koje se temelje na gradijentima i bez gradijenata. Metode temeljene na gradijentu započinju s preslikavanjem iz konačno-dimenzionalnog prostora u prostor pravila, s obzirom na prametar vektora:

$$\rho(\theta) = \rho^{\pi\theta}$$

Formula 9. Parametar vektora

U blagim uvjetima će ova funkcija biti diferencirana kao funkcija parametra vektora. Ako je gradijent poznat, mogla bi se koristiti metoda uspona gradijenta. Budući da analitički izraz za gradijent nije dostupan, dostupna je samo procjena. Takva se procjena može konstruirati na više načina, stvarajući algoritme poput Williamsove reinforce metode(metoda omjera vjerojatnosti u literaturi za optimizaciju na temelju simulacije). Metode pretraživanja pravila korištene su u kontekstu robotike. Mnoge metode pretraživanja pravila mogu se zaglaviti u lokalnoj optimi jer se temelje na lokalnoj pretrazi. Veliki razred metoda izbjegava oslanjanje na informacije o gradijentu. Oni uključuju simulirano žarenje, unakrsnu entropiju pretraživanja ili metode evolucijskog izračuna. Mnoge metode bez gradijenta mogu postići (u teoriji i u granicama) globalni optimum. Načini pretraživanja pravila mogu se polako konvergirati s obzirom na podatke. Ponašanje asimptota i konačni uzorak većine algoritama je dobro razumljiv. Poznati su algoritmi s dobrim učinkom na mreži koji rješavaju problem istraživanja. Učinkovito istraživanje velikih MDP-a u velikoj je mjeri neistraženo. Iako su se za mnoge algoritme pojavile granice vremenskih ograničenja, očekuje se da će ove granice biti prilično labave, pa će trebati više rada da bi se bolje razumjele relativne prednosti i ograničenja. Za inkrementalne algoritme riješeni su problemi asimptotske konvergencije. Algoritmi temeljeni na vremenskim razlikama konvergiraju se pod širim setom uvjeta nego što je to prije bilo moguće (na primjer, kada se koriste s proizvoljnom, glatkom aproksimacijom funkcije).

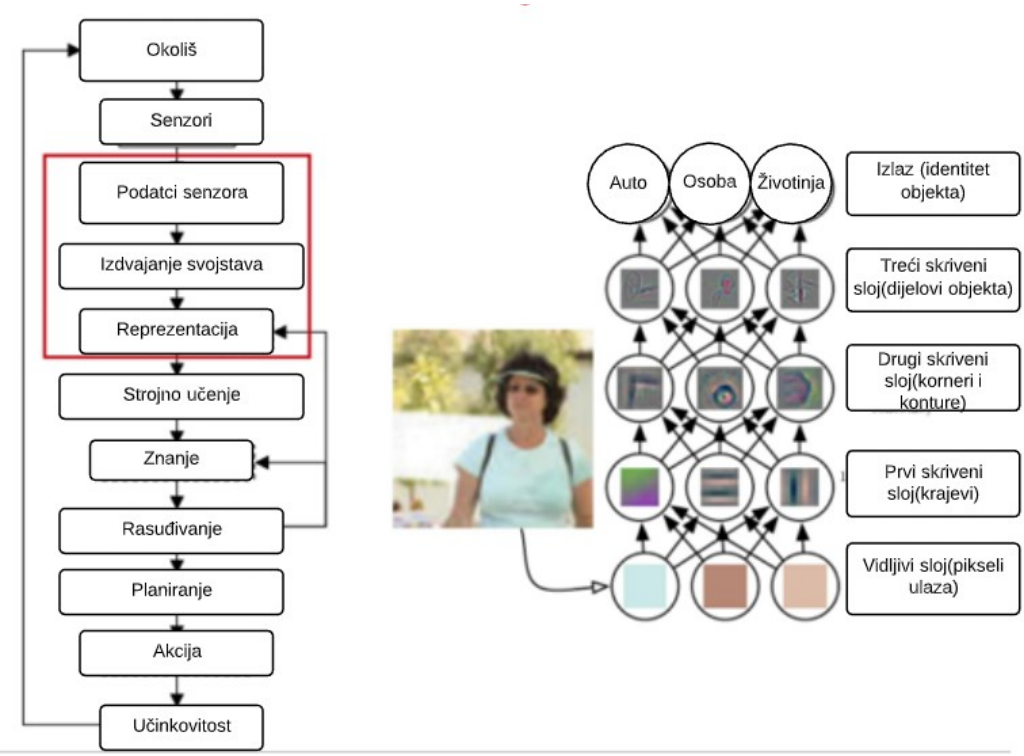
Algoritam	Opis	Model	Pravilo	Prostor akcije	Prostor stanja	Operator
Monte Carlo	Svako posjećivanje metode Monte Carlo	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Uzorak sredstva
Q-learning	Stanje-akcija-nagrada-stanje	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Q-vrijednost
SARSA	Stanje-akcija-nagrada-stanje-akcija	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Q-vrijednost

Q-learning-Lambda	Stanje-akcija-nagrada-stanje sa tragovima kvalificiranosti	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Q-vrijednost
SARSA-Lambda	Stanje-akcija-nagrada-stanje-akcija sa tragovima kvalificiranosti	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Q-vrijednost
DQN	Duboki Q mreža	Bez modela	Neovisan o akcijama agenta	Diskretan	Diskretan	Q-vrijednost
DDPG	Duboki deterministički gradijenti pravila	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Q-vrijednost
A3C	Asinkrona prednost algoritma glumca-kritičara	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Prednost
NAF	Q-učenje sa normaliziranim prednostima funkcija	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Prednost
TRPO	Optimizacija regija istine pravila	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Prednost
PPO	Proksimalna optimizacija pravila	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Prednost
TD3	Dvostruko odgođeni duboki deterministički gradijent pravila	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Q-vrijednost
SAC	Meki glumac-kritičar	Bez modela	Neovisan o akcijama agenta	Kontinuiran	Kontinuiran	Prednost

Tablica 10. Usporedba algoritama strojnog podržanja

3 Odabir domene za učenje podržanja

Jedan od načina da se zamisle autonomna sredstva za učenje podržanja bilo bi u primjeru slijepe osobe koja pokušava ploviti svijetom samo ušima i bijelim štapom. Agenti imaju male prozore koji im omogućavaju da opažaju svoje okruženje, a ti prozori možda čak i nisu najprikladniji način da shvate što je oko njih. Teško je riješiti odluku na koje vrste unosa i povratnih informacija treba reagirati agent, to se zove odabir domene. Algoritmi koji uče kako igrati video igre uglavnom ovaj problem ignoriraju jer je okruženje umjetno i strogo ograničeno. Stoga video igre pružaju sterilno okruženje laboratorija u kojem se mogu testirati ideje o očenju podržanja. Odabir domena zahtijeva ljudske odluke, obično na temelju znanja ili teorija o problemu koji se rješava; npr. odabir domena unosa algoritma u automobilu za samostalno upravljanje koji može uključivati odabir da se dodaju radarski senzori, osim kamera i GPS podataka.



Slika 11. Deep Learning za aute koji voze sami

4 Parovi stanja-akcije i složene raspodjele vjerojatnosti nagrade

Cilj učenja podržanja je odabrati najbolje poznate akcije za bilo koje stanje, što znači da se akcije moraju rangirati i dodijeliti vrijednostima da vrijednosti budu jedna u drugoj. Budući da su te akcije ovisne o stanju, ono što mi stvarno procjenjujemo je vrijednost parova između stanja i akcije; tj.radnja poduzeta iz određenog stanja, nešto što smo negdje učinili. Postoje nekoliko primjera koji pokazuju da je vrijednost i značenje radnje ovisna o stanju u kojemu je ta radnja poduzeta : 1. Ako se radnja udaje za nekoga, tada brak s 35-godišnjakom kada imate 18 godina vjerojatno znači nešto drugačije od udaje za 35-godišnjaka kada imate 90 godina, a ta dva ishoda vjerojatno imaju različite motivacije i dovode do različitih rezultata, 2. Ako radnja viče „Vatra!“, izvođenje akcije gdje je kazalište prepuno trebalo bi značiti nešto drugačije od izvođenja akcije pored ljudi s puškama. Ne možemo predvidjeti ishod akcije bez poznavanja konteksta. Mapiramo parove stanja-akcije u vrijednosti za koje očekujemo da će ih proizvesti pomoću Q funkcije, jer ona uzima kao svoj ulaz agenta, stanje i djelovanje te ih preslikava na moguće nagrade. Učenje podržanja proces je pokretanja agenta kroz nizove parova stanja-akcije, promatranje nagrada koje rezultiraju i prilagođavanje predviđanja Q funkcije tim nagradama dok se točno ne predvidi najbolji put za agenta, takvo predviđanje poznato je kao pravilo. Strojno podržanje je u principu pokušaj modeliranja složene distribucije vjerojatnosti nagrada u odnosu na vrlo veliki broj parova stanja-akcija. To je jedan od razloga što je učenje podržanja povezano s Markovim postupkom odlučivanja, metodom uzorkovanja iz složene distribucije da bi se zaključila njena svojstva. Svaki statistički pristup u osnovi je priznanje neznanja. Ogromna složenost nekih pojava (bioloških, političkih, sociooloških ili povezanih s igrama na ploči) onemogućuje rasuđivanje iz prvih načela. Jedini način da ih proučimo je kroz statistiku, mjerenjem površnih događaja i pokušaj uspostavljanja povezanosti među njima, čak i kad ne razumijemo mehanizam na koji se oni odnose. Strojno podržanje poput dubokih neuronskih mreža, jedna je od takvih strategija koja se oslanja na uzorkovanje za izvlačenje podataka iz baze podataka. Nakon malo vremena provedenog u korištenju Markovskog procesa odlučivanja za približavanje vjerojatnosti raspodjele nagrade nad parovima stanja djelovanja, algoritam učenja za

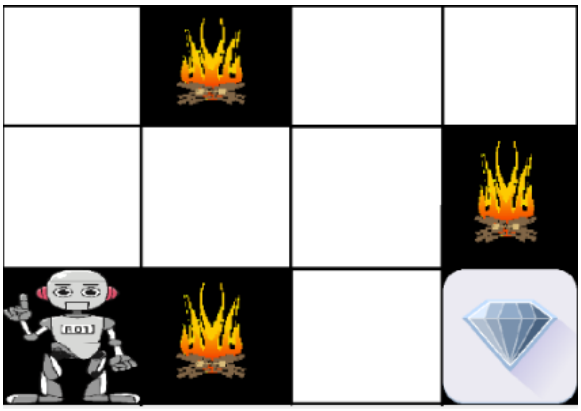
podržanje može imati tendenciju ponavljanja radnji koje dovode do nagrade i prestaju testirati alternative.

Postoji napetost između iskorištavanja poznatih nagrada i nastavka istraživanja kako bi se otkrili novi postupci koji također vode do pobjede. Baš kao što naftne kompanije imaju dvostruku funkciju crpljenja sirove nafte iz poznatih naftnih polja dok buše nove rezerve, tako se i algoritmi za učenje podržanja mogu načiniti, iskorištavati i istraživati u različitoj mjeri kako bi se osiguralo da one ne prođu preko akcija nagrađivanja na štetu poznatih pobjednika, time spriječavajući da loš ishod prevlada dobar ishod. Učenje podržanja je iterativno. U svojim najzanimljivijm primjenama ne započinje znanjem koje će nagrade proizvesti parovi stanja-akcija. Uči te odnose trčeći kroz stanja stalno i iznova, poput sportaša ili glazbenika, ponavljajući stanje u pokušaju da nakon svake iteracije poboljša svoj učinak.

4.1 Odnos strojnog učenja s vremenom

Moglo bi se reći da je algoritam metoda bržeg objedinjavanja lekcija vremena. Algoritmi za strojno podržanje učenja imaju različit odnos prema vremenu nego ljudi. Algoritam se može ponavljati kroz ista stanja iznova i iznova eksperimentirajući s različitim radnjama, dok ne može zaključiti koje su akcije najbolje iz kojih stanja. Učinkovito, algoritmi uživaju u svom vlastitom vremenskom periodu gdje počinju kao glupi i postepeno se opamećuju. Budući da ljudi nikada ne dožive ovako nešto izvan filma, algoritmi za učenje podržanja imaju potencijal da nauče više i bolje od ljudi. Doista, istinska prednost ovih algoritama nad ljudima ne proizlazi toliko iz njihove inherentne prirode, koliko od njihove sposobnosti da paralelno žive na mnogim čipovima odjednom, da treniraju noć i dan bez umoran, a samim tim i da nauče više. Algoritam obučen u igri Go, kao što je AlphaGo, odigrat će mnogo više Go igara nego što se bilo koji čovjek može nadati da će ga dovršiti u 100 života, ista je priča i za kompleksnije igre poput Starcrafta ili Dote, gdje će algoritam konstantno učiti na svojim greškama i memorirati što treba raditi, a što ne treba, puno bolje nego čovjek. Jednostavan primjer za ovako nešto je kada imamo problem gdje

imamo agenta i nagradu, s mnogim preprekama između. Agent bi trebao pronaći najbolji mogući put do nagrade.

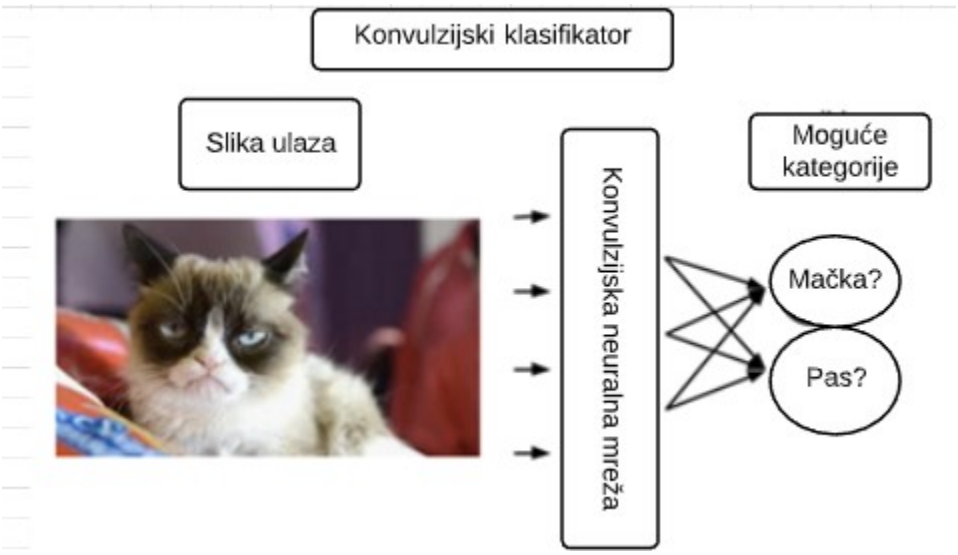


Slika 12. Robot,prepreka i nagrada

Gornja slika prikazuje robota, dijamante i vatru. Cilj robota je dobiti nagradu koja je dijamant i izbjeći prepreke u obliku vatre. Robot uči iskušavajući sve moguće staze i tada bira put koji mu daje nagradu s najmanje prepreka. Svaki pravi korak će robotu omogućiti nagradu, a svaki pogrešan korak oduzet će mu nagradu. Ukupna nagrada će se izračunati kada dostigne konačnu nagradu u obliku dijamanta. Ono što je je važno za shvatiti je da ovako nešto ne bismo mogli postići sa nadzornim učenjem, zbog znatnih razlika između nadzornog i strojnog učenja podržanja. Učenje podržanja odnosi se na redoslijed donošenja odluka. Jednostavnim riječima može se reći da izlaz ovisi o stanju trenutnog ulaza, a sljedeći ulaz ovisi o izlazu prethodnog ulaza. U procesu podržanja odluka o učenju daju se oznake nizovima ovisnih odluka, primjerice u partiji šaha. U nadzornom učenju odluka se donosi na početnom unosu ili unosu danom na početku. Odluke o nadzoru učenja neovisne su jedna o drugoj, tako da se za svaku odluku daju oznake, npr. prepoznavanje predmeta. Time možemo zaključiti da učenje podržanja se razlikuje od nadziranog učenja na način da u nadgledanom učenju podaci imaju tipku odgovora, tako da se model trenira s točnim odgovorom, dok u učenju podržanja nema odgovora, nego agent za podržanje odlučuje što učiniti za izvršavanje zadanog zadatka. U nedostatku podataka o obuci, dužan je učiti iz svog iskustva.

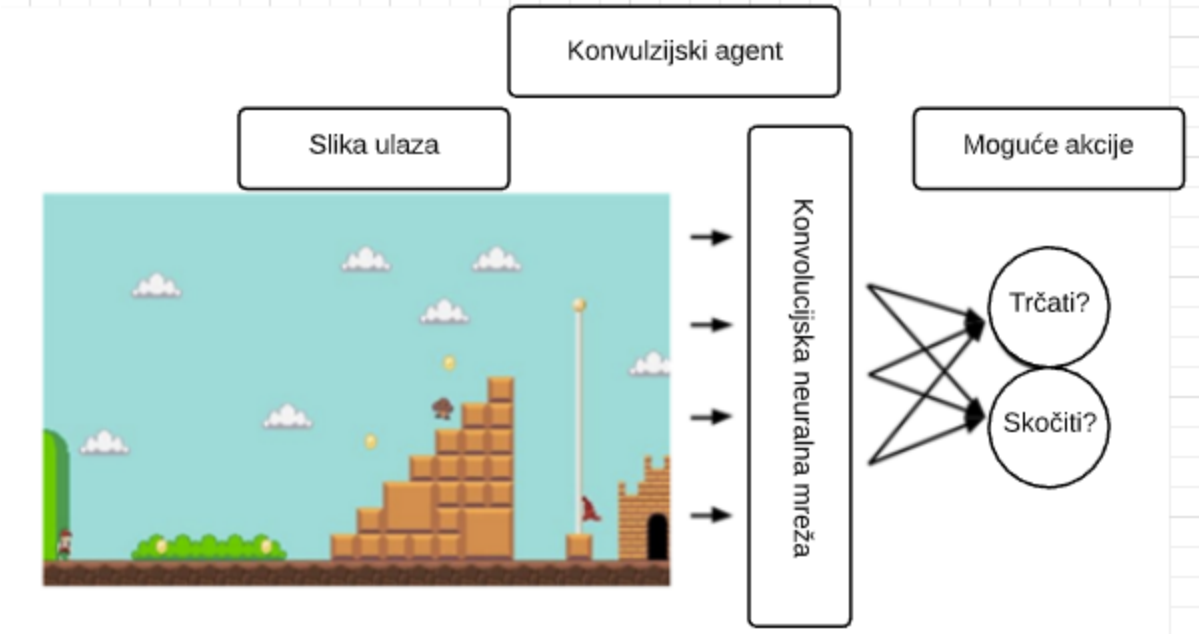
5 Neuronske mreže i duboko podržanje učenja

Neuronske mreže su sredstvo koje uči mapirati parove stanja i akcije u nagradu. Kao i sve neuronske mreže, oni koriste koeficijente za približavanje funkcije koja se odnosi na ulaze i izlaze, a njihovo se učenje sastoji od pronalaženja pravih koeficijenata ili utega, tako da iterativno podešavaju utege duž gradijenata koji obećavaju manju grešku. Kod učenja podržanja mogu se koristiti revolucionarne mreže za prepoznavanje stanja agenata; npr. ekran na kojem je Mario, ili teren prije drona. Odnosno, oni obavljaju svoj tipični zadatak prepoznavanja slike. Konvolucionarne mreže dobivaju različita tumačenja od slika u podržanju učenja, nego u učenju pod nadzorom. U nadzornom učenju mreža primjenjuje oznaku na sliku, točnije odgovara imenima u pikselima.



Slika 13. Konvulzijski klasifikator

Algoritam će rangirati naljepnice koje najbolje odgovaraju slici u pogledu njihovih vjerojatnosti. Prikazana slika mačke može algoritmu omogućiti da odluči da je slika 80% vjerojatno mačka, 50% vjerojatno tigar i 30% vjerojatno pas. U strojnom podržanju učenja imajući na umu ako imamo sliku koja predstavlja nekakvo stanje, mreža zavojnice može rangirati radnje koje je moguće izvesti u tom stanju; na primjer, može se predvidjeti da će trčanje udesno vratiti 5 bodova, skok 7 i trčanje ulijevo nijedan.



Slika 14. Konvulzijski agent

Gornja slika prikazuje što čini agent pravila, mapirajući stanje u najbolju radnju.

$$Q(s_t, a_t) \Leftarrow (1-\alpha) \cdot Q(s_t, a_\tau) + \alpha \cdot (r_t + \gamma \cdot \max Q(s_t + 1, a))$$

Formula 15. Formula za najveću kombinaciju momentalnih nagrada i budućih

Dodjeljujući vrijednosti očekivanim nagradama, funkcija Q jednostavno odabire par stanja radnje s najvišom takozvanom Q vrijednošću. Na početku učenja podržanja, koeficijenti neuronske mreže mogu se inicijalizirati stohastički ili nasumično. Koristeći povratne informacije iz okoline, neuralna mreža može upotrijebiti razliku između očekivane nagrade i nagrade na temelju istine da bi prilagodila svoje težine i poboljšala interpretaciju parova stanja-akcija. Ova petlja za povratne informacije analogna je u uzvraćanju pogrešaka kod nadziranog učenja. Međutim nadzirano učenje započinje poznavanjem oznaka temeljne istine koje neuronska mreža pokušava predvidjeti. Njegov je cilj stvoriti model koji preslikava različite slike na njihova imena. Učenje podržanja oslanja se na okoliš i šalje mu skalarni broj kao odgovor na svaku novu akciju. Nagrade koje okolina vraća mogu se mijenjati, odgađati ili utjecati na nepoznate varijable, unoseći buku u povratnu petlju.

To dovodi do cjelovitijeg izražavanja Q funkcije, koje uzima u obzir ne samo trenutne nagrade proizveden djelovanjem, već i odgođene nagrade koje se mogu vratiti nekoliko

vremenskih koraka dublje u nizu. Kao i ljudska bića, Q funkcija je rekurzivna. Baš kao što nazivanje metode software-a human() sadrži u sebi još jednu metodu human(), od koje smo svi mi plod, tako i pozivanje Q funkcije na određeni par stanje-akcija zahtijeva da nazovemo ugniježđenu Q funkciju kako bismo predvidjeli vrijednost sljedećeg stanja, koje zauzvrat ovisi o Q funkciji stanja nakon toga, itd. Bilo bi korisno zamisliti algoritam učenja podržanja u akciji, kako bi ga vizualno oslikali. Recimo da algoritam uči da igra videoigru Super Mario. Mario pokušava kroz igru steći najviše bodova. Da bismo to postigli, možemo paralelno vrtiti mnoštvo različitih Maria i pokretati ih kroz prostor svih mogućih stanja igre. Kao da imate 1000 Maria koji prolaze kroz planinu, i dok kopaju (npr. dok odlučuju koju će radnju poduzeti kako bi utjecali na okruženje igre), njihovi se tuneli doživljavaju poput zamršenih i fraktalnih grančica stabla. Mariovi tuneli su hodnici svjetlosti koji se probijaju kroz planinu. I kao i u samom životu, jedna uspješna akcija može učiniti vjerojatnijim da je uspješna akcija moguća u većem protoku odluka, što će potaknuti pobjedničkog Maria da ide dalje. Možemo zamisliti, ako je svaki Mario agent, ispred njega je toplinska karta koja prati nagrade koje može povezati s parovima stanje-akcija. Zamislimo da svaki par radnji u stanju ima vlastiti ekran prekriven toplinom od žute do crvene boje. Mnogi ekrani su sastavljeni u rešetku, kao što je vidljivo i u nekakvoj firmi koja se bavi server administracijom. Jedan zaslon akcije mogao bi biti „Teže skočite u ovom stanju“, drugi bi mogao biti „trči brže u ovom stanju“ itd.. Budući da neki parovi stanja-akcija dovode do znatno veće nagrade od drugih, i različite vrste akcija kao što su skakanje, čučnjevi ili trčanje, možemo uzeti raspodjelu vjerojatnosti nagrade nad akcijama. Budući da algoritam počinje neupućen i mnogi su putevi kroz prostor stanja igre neistraženi, toplotne mape će odražavati njihov nedostatak iskustva; tj. može se dogoditi praznina u toplinskoj mapi. Super Mariji su u osnovi rakete za traženje nagrade koje vode te toplotne karte, a što više puta prolaze kroz igru, njihov toplotni plan potencijalne buduće nagrade postaje točniji. Toplinske karte su u osnovi vjerojatne raspodjele nagrada nad parovima stanje-akcija.

6 Deepmind

Igre se već desetljećima koriste kao važan način za testiranje i procjenu performansi sustava umjetne inteligencije. Kako su se mogućnosti povećavale, istraživačka zajednica tražila je igre sa sve većom složenošću koje obuhvaćaju različite elemente inteligencije potrebne za rješavanje znanstvenih i stvarnih problema. Posljednjih godina, Starcraft, koji se smatra jednom od najizazovnijih strategija u stvarnom vremenu (RTS) i jednim od najdugovječnijih izvođenja svih vremena, konsenzusom se pojavio kao „veliki izazov“ za AI istraživanje. Sada se predstavlja StarCraft 2 program koji se zove AlphaStar, prvu umjetnu inteligenciju koja je porazila vrhungskog profesionalnog igrača. U nizu testnih mečeva koji su održani 19. prosinca, AlphaStar je odlučno pobijedio Grzegorza “MaNa” Komincza koji igra za tim Liquid, 5-0, nakon uspješnog referentnog meča protiv njegovog suigrača Daria “TLO” Wunsch. Utakmice su se odvijale u uvjetima profesionalnih utakmica na natjecateljskoj karti ljestvice bez ikakvih ograničenja igre. Iako je došlo do značajnih uspjeha u video igrama kao što su Atari, Mario, Quake III Arena pa čak I Dota 2, do sada su se AI tehnike teško borile sa složenošću StarCrafta. Najbolji rezultati omogućeni su ručnim izrađivanjem glavnih elemenata sustava, nametanjem značajnih ograničenja pravilima igre, davanjem sustava nadljudskih mogućnosti ili igranjem na pojednostavljenoj mapi. Čak ni uz ove izmjene, niti jedan sustav nije se približio kvalifikaciji vještina profesionalnih igrača. Suprotno tome, AlphaStar igra cjelokupnu igru StarCraft II, koristeći duboku neuronsku mrežu koja se izučava izravno iz sirovih podataka o igri pod nadzorom učenja i učvršćenja.

6.1 Izazov StarCrafta

StarCraft II, kreiran od strane tvrtke Blizzard Entertainment, postavljen je u izmišljeni znanstveno-fantastični svemir i sadrži bogat, višeslojni gameplay osmišljen da izazove

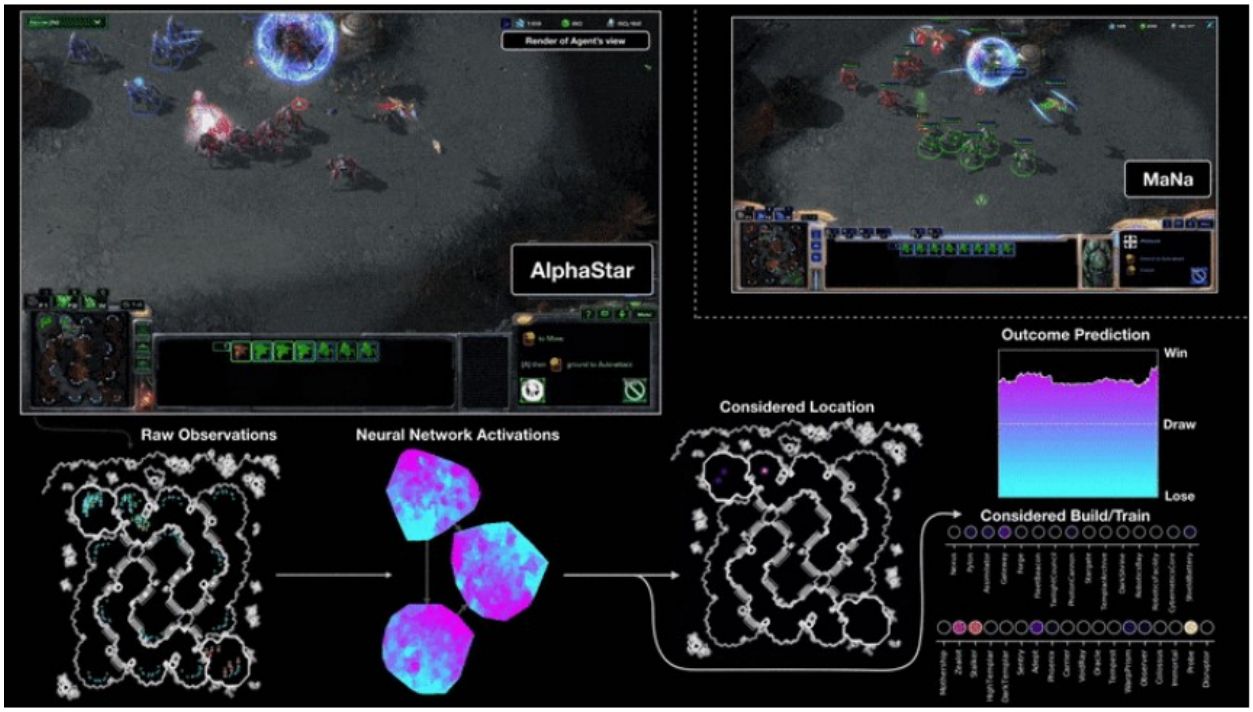
ljudski intelekt. Uz originalni naslov, spada među najveće i najuspješnije igre svih vremena, a igrači se natječu u esports turnirima više od 20 godina.

Slika 16. Izgled StarCrafta sa strane posrednika koji ima pogled na resurse AlphaStara I TLO-a

Postoji nekoliko različitih načina igranja, ali u esportovima najčešći je turnir 1v1 koji se igra na pet utakmica. Za početak, igrač mora izabrati jednu od tri različite vanzemaljske “rase” – Zerg, Protoss ili Terran, koje imaju sve karakteristike i sposobnosti (iako se profesionalni igrači teže specijalizirati za jednu rasu). Svaki igrač započinje s brojnim radničkim jedinicama koje prikupljaju osnovne resurse za izgradnju više jedinica i struktura i stavaranje novih tehnologija. Oni zauzvrat omogućuju igraču da prikupi druge resurse, izgradi sofisticiranije baze i strukture i razvije nove mogućnosti koje se mogu iskoristiti za nadmudrivanje protivnika. Da bi pobijedio, igrač mora pažljivo uravnotežiti upravljanje velikim mehanikama svoje ekonomije – poznato pod nazivom makro – zajedno s niskom razinom kontrole svojih pojedinačnih jedinica – poznatih kao mikro. Potreba za uravnoteženjem kratkoročnih i dugoročnih ciljeva te prilagođavanje neočekivanim situacijama predstavlja veliki izazov za sustave koji su često bili krhki i nefleksibilni. Za savladavanje ovog problema potrebno je probiti nekoliko izazova AI

istraživanja, uključujući teoriju igre, nesavršenu informaciju, dugoročno planiranje, stvarno vrijeme i veliki prostor akcije.

- Teorija igre : StarCraft je igra u kojoj, baš kao i „škare, kamen, papir“, ne postoji nijedna najbolja strategija. Kao takav, AI proces obuke treba kontinuirano istraživati i proširiti granice strateškog znanja.
- Nesavršena informacija : Za razliku od igara poput šaha ili igara u kojima igrači sve vide, ključne se informacije kriju od igrača StarCraft-a i moraju ih aktivno otkrivati izviđači.
- Dugoročno planiranje : Kao i mnogi problem iz stvarnog svijeta, uzrok posljedica nisu trenutačni. Radnje poduzete u ranoj fazi igre, neće se dugoročno isplatiti.
- Stvarno vrijeme : Za razliku od tradicionalnih igara na ploči u kojima igrači izmjenjuju poteze između sljedećih poteza, StarCraft igrači moraju neprekidno izvoditi radnje kako vrijeme napreduje.
- Veliki akcijski prostor : Stotine različitih jedinica i zgrada moraju se kontrolirati odjednom, u stvarnom vremenu, što rezultira kombinatornim prostorom mogućnosti. Povrh svega, akcije su hijerarhijske i mogu se mijenjati te nadopunjavati. Deepmind parametrizacija igre u prosjeku ima otprilike 10 do 26 pravnih radnji u svakom koraku. Zbog ovih ogromnih izazova, StarCraft je postao “veliki izazov” za AI istraživanje. Natjecanja koja su u tijeku i u StarCraftu i u StarCraft II ocjenjivala su napredak od pokretanja BroodWar API-ja 2009. Godine, uključujući AIIDE StarCraft AI natjecanje, CIG StarCraft natjecanje, Student Starcraft AI turnir I StarCraft II AI ljestvicu. Da se zajednici pomogne, Deepmind je radio s Blizzardom u 2016. i 2017. godini na izdavanju skupa alata otvorenog koda koji su poznati kao PySC2, uključujući najveći set anonimnih ponavljanja igara koje su ikada objavljene. Kombiniranjem inženjeringa i algoritamske revolucije se izgradio posao za proizvodnju AlphaStara.



Slika 17. Vizualizacija AlphaStar agenta u borbi protiv profesionalnog igrača MaNe

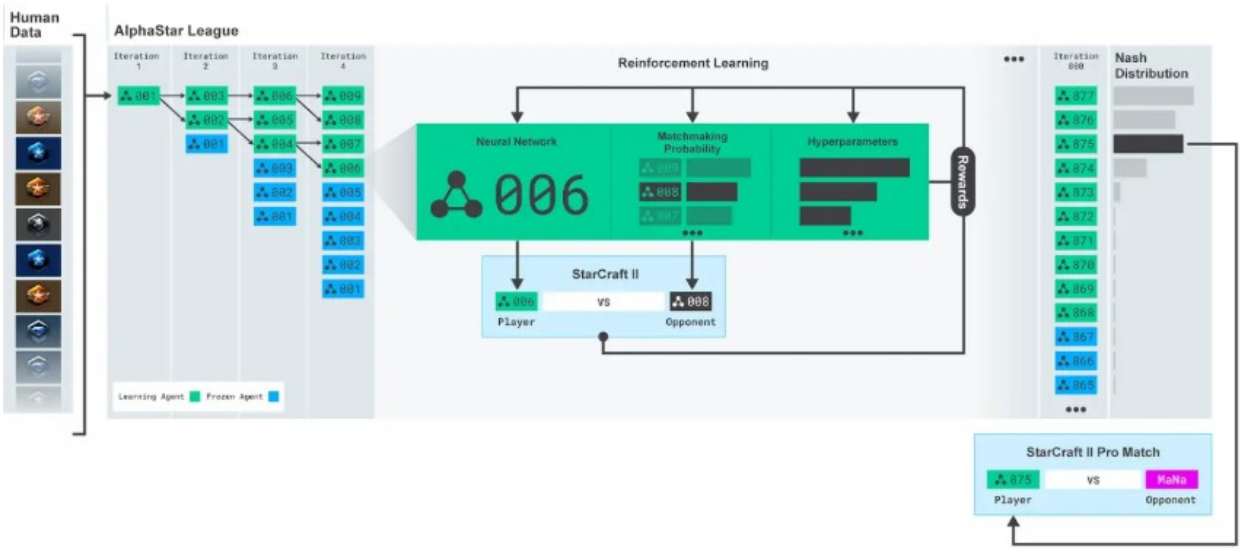
Gornja slika pokazuje utakmicu sa sredstvom gledišta koje zovemo sirovo promatranje ulaza na neuronske mreže i neuronske mreže internih aktivacija. Neke od aktivnosti koje agent može koristiti odnose se na opcije gdje kliknuti i što graditi, a predviđene su ishodom.

6.2 Kako se trenira AlphaStar

Ponašanje AlphaStara generira duboka neuronska mreža koja prima ulazne podatke iz surovog sučelja igre (popis jedinica i njihovih svojstava) i iskazuje niz uputa koje čine radnju u igri. Konkretnije, arhitektura neuronske mreže primjenjuje torzo transformatora na jedinice (slično kao relacijsko učenje s dubokim ojačanjima), kombinirano s dubokom LSTM jezgrom, auto-regresivnom glavom pravila s mrežom, pokazivača i središnjom vrijednosti. Vjeruje se da će ovaj napredni model pomoći u mnogim drugim izazovima u

istraživanju strojnog učenja koji uključuju dugoročno modeliranje sekvenci i velike izlazne prostore kao što su prijevod, jezično modeliranje i vizualni prikaz. AlphaStar također koristi novi algoritam učenja s više agenata. Neuralna mreža bila je u početku obučena pod nadzorom učenja iz anonimnih ljudskih igara koje je objavio Blizzard.

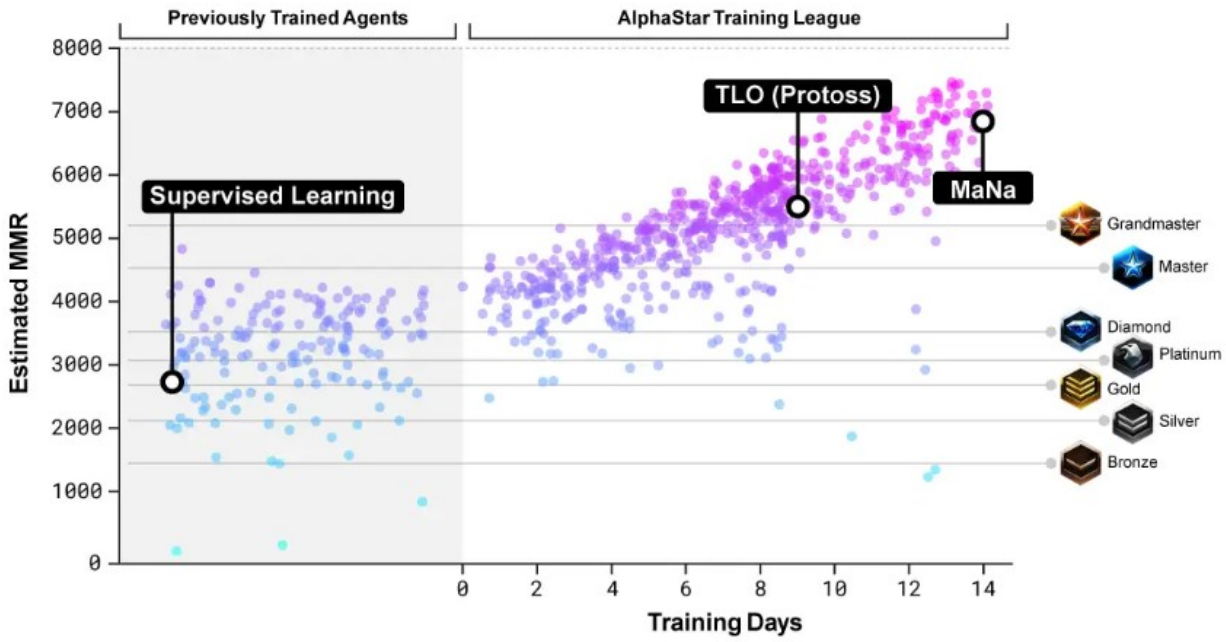
To je AlphaStaru omogućilo da, imitacijom, nauči osnovne mikro i makro strategije koje igrači koriste na StarCraft ljestvici. Ovaj početni agent porazio je ugrađeni AI nivo „Elite"-oko razine zlatne lige za ljudskog igrača – u 95% igara.



Slika 18. AlphaStar agenti koji uče na ljudskim replayovima utakmica

Agenti su inicijalno trenirani od prijenosa utakmica koje su igrali pravi ljudi te su onda trenirali jedan protiv drugog. U svakoj iteraciji, novi su svedeni u grane, originalni natjecatelji su zamrznuti, a proizvodi i hiperparametri određuju učinkoviti cilj za svakog agenta. Agent se može prilagoditi, povećavajući različitost svojih sesija. Parametri agenata su ažurirani strojnim podržanjem koje se izučavalo protiv natjecatelja. Završni agent je postavljen kao uzorak bez zamjene tokom distribucije lige. Zatim su korišteni za podučavanje procesa učenja s višestrukim agentima. Stvorena je kontinuirana liga, u kojoj su agenti lige -natjecatelji, igrali jedni protiv drugih, slično načinu na koji ljudi doživljavaju igru StarCrafta igrajući StarCraft ljestvicu. U konkurenciju su se dinamički dodavali novi natjecatelji grananjem s postojećim natejcateljima, svaki agent tada uči od

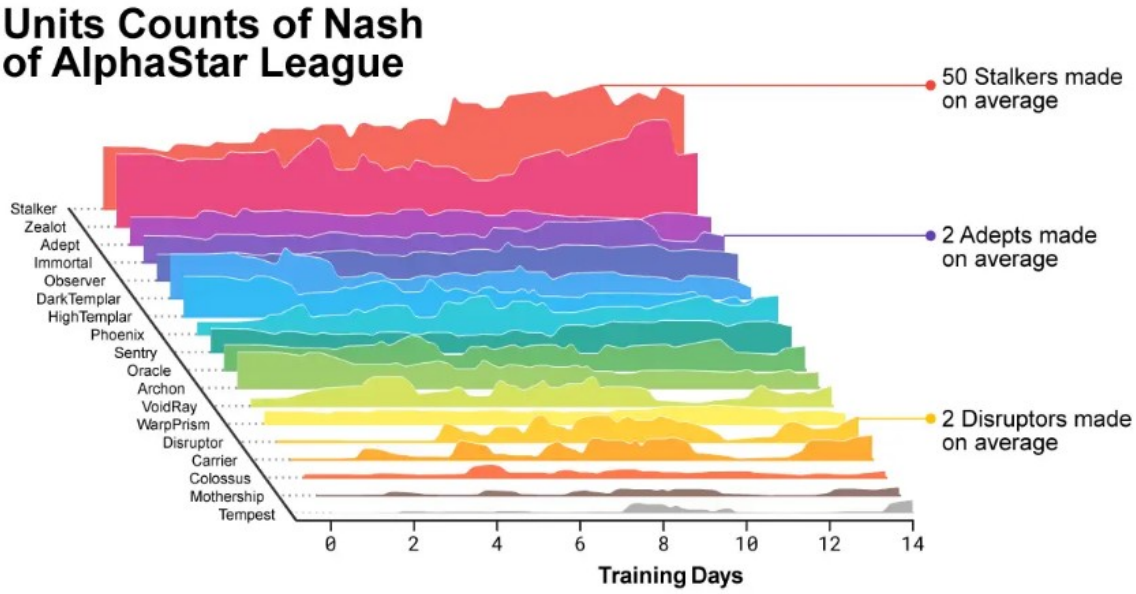
igara protiv drugih natjecatelja. Ovaj novi oblik treninga dodatno preuzima ideju podržanog stanovništa i multiagencije, svarajući proces koji kontinuirano istražuje ogroman strateški prostor StarCraft gameplay-a, istovremeno osiguravajući da svaki natjecatelj djeluje dobro protiv najjačih strategija, pritom ne zaboravlja kako pobijediti ranije.



Slika 19. Prikazuje aproksimativno potrošeno vrijeme za prosječno iskustvo ili MMR određene lige

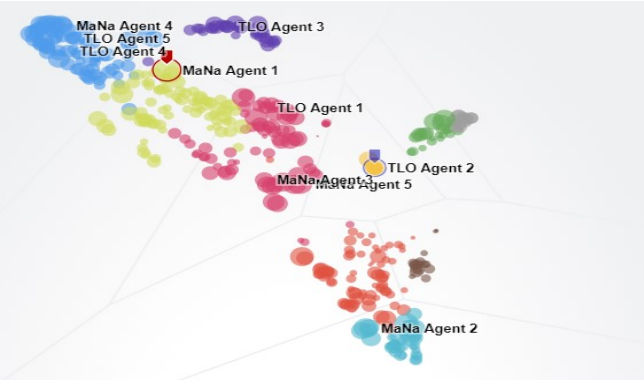
Kako liga napreduje i stvaraju se novi natjecatelji, pojavljuju se nove kontra-strategije koje su u stanju pobijediti ranije strategije. Dok neki novi konkurenti izvršavaju strategiju koja je samo usavršavanje prethodne strategije, drugi otkrivaju drastično nove strategije koje se sastoje od potpuno novih naloga za izgradnju, sastava jedinica i planova mikro upravljanja. Na primjer, u ranoj fazi AlphaStar lige favorizirane su “sirevne” strategije poput brzog trčanja s Photon Cannonima ili Dark Templarsima. Ove rizične strategije odbačene su kako se obuka razvijala, što je dovelo i do drugih strategija : na primjer, stjecanja ekonomske snage prekomjernim proširenjem baze s više radnika ili žrtvovanjem dviju oracle-a kako bi poremetili protivničke radnike i ekonomiju. Ovaj je postupak sličan

načinu na koji su igrači otkrili nove strategije i uspjeli pobijediti prethodno favorizirane pristupe tijekom godina od izdavanja StarCrafta.

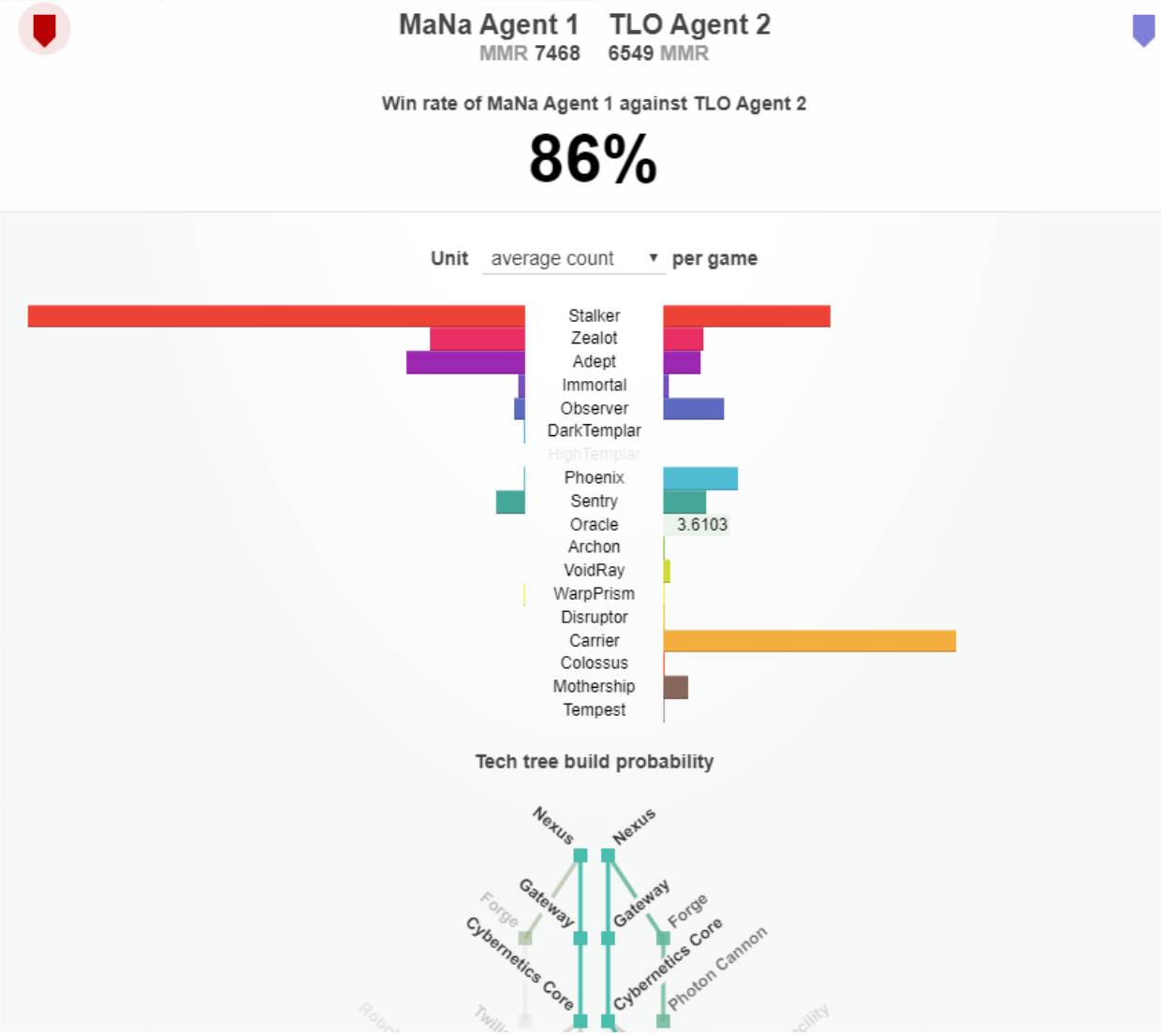


Slika 20. Prikaz adaptacije građenja jedinica AlphaStara tokom mijenjanja lige

Kako bi se potaknula raznolikost u ligi, svaki agent ima svoj vlastiti cilj učenja : na primjer, koji natjecatelji trebaju ovo sredstvo pobijediti i sve dodatne unutarnje motivacije koje utječu na to kako agent igra. Jedan agent može imati cilj pobijediti jednog određenog konkurenta, dok drugi agent mora pobijediti čitavu raspodjelu natjecatelja, ali to može napraviti tako što će izgraditi više određene igre. Ovi ciljevi učenja prilagođeni su tijekom obuke.



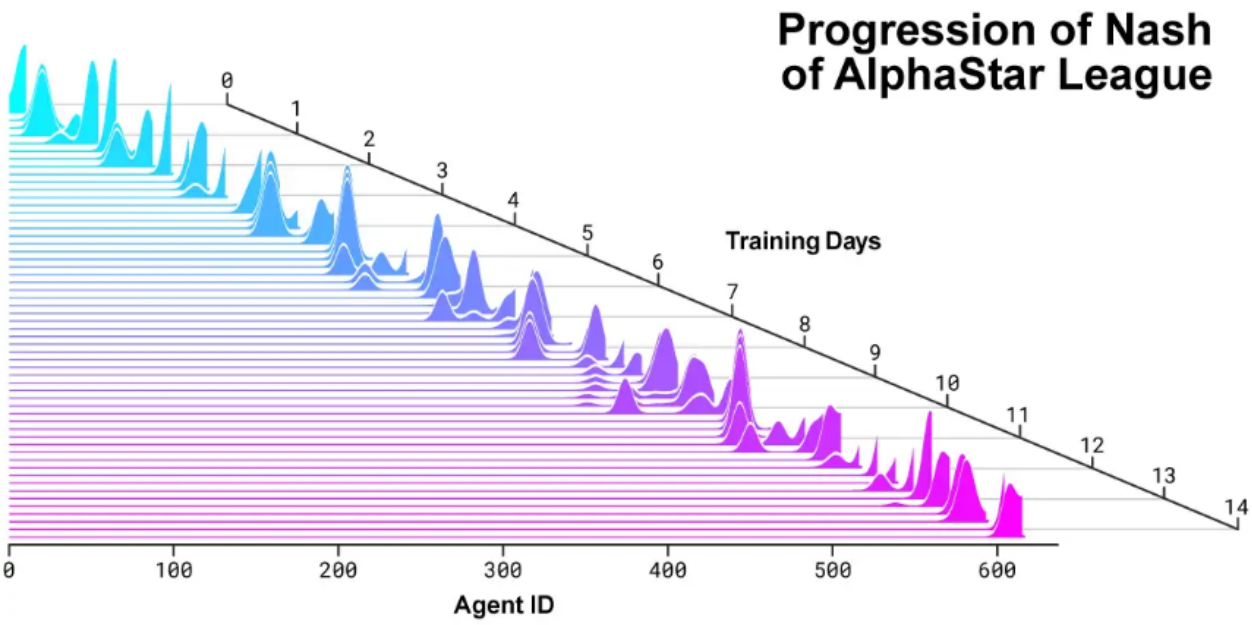
Slika 21. Prikaz klasterirane strategijske mape AlphaStara



Slika 22. Prikaz interaktivne vizualizacije koja pokazuje kompeticiju AlphaStar lige. Prikazuje agente koji su igrali protiv TLO-a i MaNe

Težine neuronske mreže svakog agenta ažuriraju se podržanjem učenja iz njegovih igara protiv konkurenata kako bi se optimizirao njegov osobni cilj učenja. Pravilo ažuriranja težine učinkovit je i novi algoritam učenja podržanja aktera i kritičara izvan pravila s ponavljanjem iskustva, samoimitacijskim učenjem i destilacijom pravila.

Kako bi se osposobio AlphaStar, potrebno je izgraditi visoku skalabilnu distribuciju postavki treninga pomoću Google-ovih v3 TPU-ova koji podržavaju populaciju agenata koji uče iz više tisuća paralelnih primjera StarCraft II. AlphaStar liga se vodila 14 dana, koristeći 16 TPU-a za svako sredstvo. Tijekom treninga, svaki agent doživio je do 200 godina u stvarnom vremenu StarCraft igre. Konačni AlphaStar agent sastoji se od komponenti Nash-ove raspodjele lige- drugim riječima, najučinkovitije mješavine otkrivenih strategija – koje se izvode na jednoj radnoj površini. Cjeloviti tehnički opis ovog rada priprema se za objavljivanje u časopisu recenziranog od strane stručanjaka.



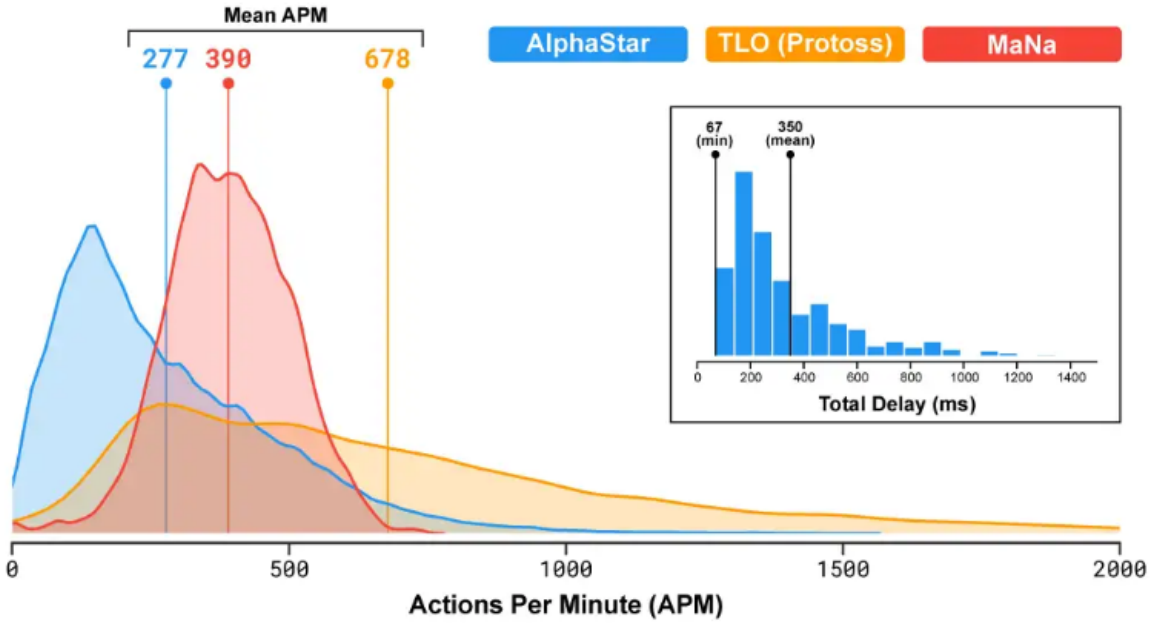
Slika 23. Pokazuje porast kompeticije AlphaStar lige kroz dane njegovog treniranja

Distribucija nasha na natjecateljima dok se AlphaStar liga širila je stvorio nove natjecatelje. Ono što je zanimljivo je da je svaki natjecatelj uvijek imao kontinuirani

napredak protiv svojih prethodnih natjecatelja, što nam govori da je AlphaStar u svakoj situaciji apsolutno uvijek učio.

6.3 Kako AlphaStar igra i nadzire igru

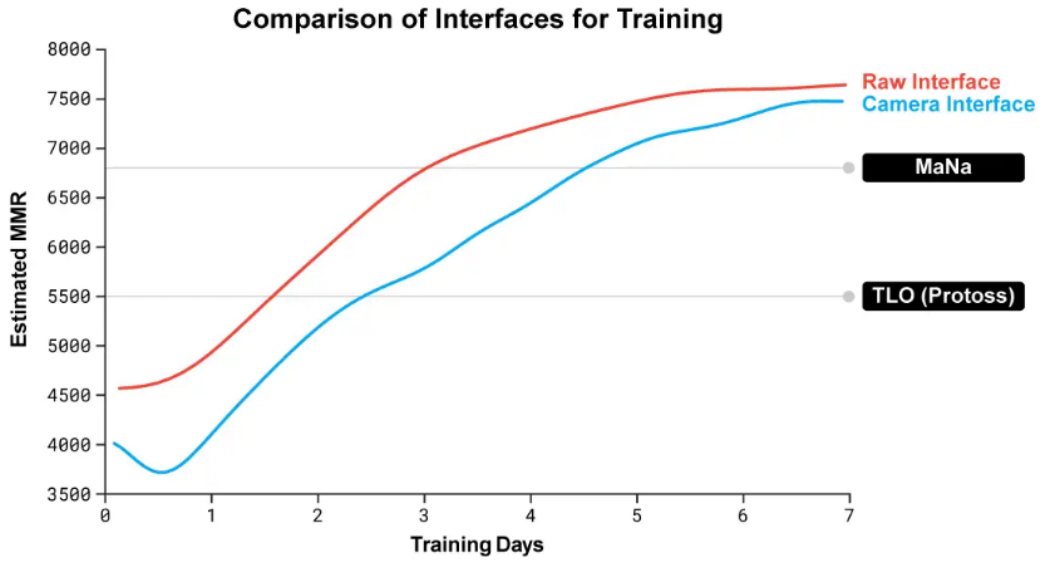
Profesionalni StarCraft igrači poput TLO i MaNa mogu u prosjeku izdati stotine akcija u minuti (APM). To je daleko manje od većine postojećih robota, koji kontroliraju svaku jedinicu neovisno i dosljedno održavaju tisuće ili čak desetke tisuća APM-ova. U igrama protiv TLO-a i MaNe, AlphaStar je imao prosječan APM od 280, znatno niži od profesionalnih igrača, iako su njegovi postupci možda precizniji. Ovaj niži APM je dijelom zato što AlphaStar započinje s treninzima pomoću ponavljanja i na taj način oponaša način na koji ljudi igraju igru. Uz to, Alphastar reagira s kašnjenje između promatranja i akcije u prosjeku od 350 milisekundi.



Slika 24. Prikaz količine akcija po minuti od AlphaStara,TLO-a i MaNe

Tijekom utakmica protiv TLO-a i MaNe, AlphaStar je izravno komunicirao s StarCraft engine-om za igre izravno kroz svoje surovo sučelje, što znači da je mogao direktno promatrati atribute vlastitih i protivničkih vidljivih jedinica na karti, bez potrebe za pomicanjem kamere – učinkovito igrajući sa uvećanim prikazom igre.

Suprotno tome ljudski igrači moraju izričito upravljati „ekonomičnošću pažnje“ kako bi odlučili gdje usmjeriti kameru. Međutim analiza AlphaStarovih igara sugerira da on upravlja implicitnim fokusom pažnje. U prosjeku, agenti su „mijenjali kontekst“ oko 30 puta u minuti, slično kao MaNa ili TLO. Uz to i nakon mečeva razvila se druga verzija AlphaStara. Kao i ljudski igrači, i ova verzija AlphaStara odabire kad i kamo premjestiti kameru, njezino je opažanje ograničeno na informacije na ekranu, a mjesta radnje ograničena su na njeno vidljivo područje.



Slika 25. Prikaz korištenja sirovog sučelja što je ekvivalentno korištenju sučelja kamere

Trenirana su dva nova agenta, jedan je koristio sirovo sučelje, a drugi mora naučiti kontrolirati kameru protiv AlphaStar lige. Svaki je agent prvotno osposobljen pod nadzorom učenja iz ljudskih podataka, nakon čega je uslijedio gore navedeni postupak

učvršćenja. Verzija AlphaStara koja koristi sučelje kamere bila je gotovo jednako jaka kao i sirovo sučelje, premašivši 7000 MMR na internoj ploči s najboljim rezultatima. U izložbenom susretu MaNa je savladao prototipsku verziju AlphaStara pomoću sučelja za kameru, koja je trenirana samo 7 dana.

U skoroj budućnosti će se vjerojatno moći procijeniti potpuno obučena instanca sučelja kamere. Ovi rezultati sugeriraju da je uspjeh AlphaStara protiv MaNe i TLO-a zapravo posljedica superiornog makro i mikro-strateškog odlučivanja, a ne superiornog broja klikova, bržeg vremena reakcije ili sirovog sučelja.

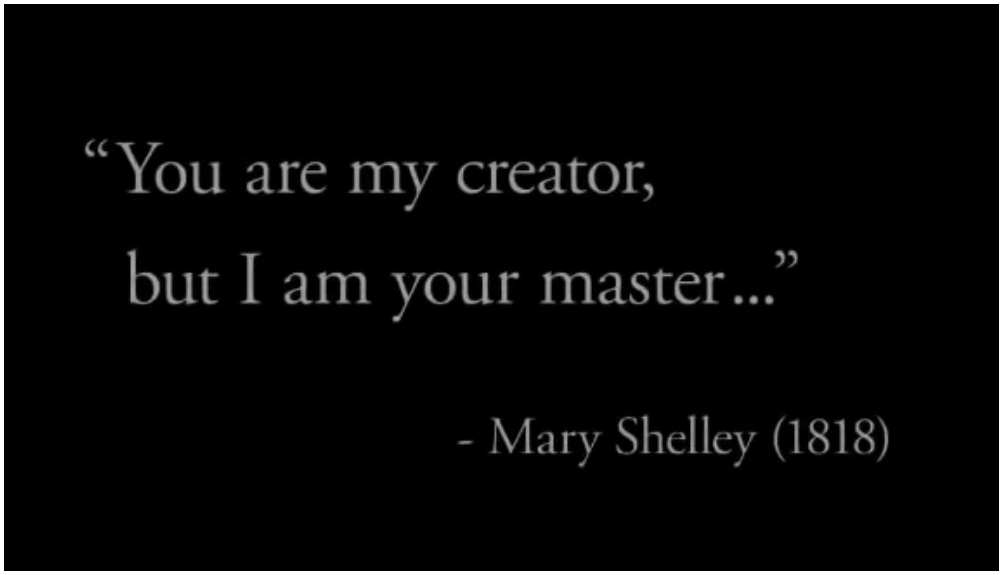
6.4 Ocjenjivanje AlphaStara prema profesionalnim igračima

Igra StarCrafta igračima omogućuje odabir jedne od tri vanzemaljske rase : Terran, Zerg ili Protoss. Za AlphaStar je izabrano da se specijalizira za igranje samo jedne rase zasad – Protoss, kako bi se smanjilo vrijeme treninga i varijante prilikom prijavljivanja rezultata iz interne lige. Isti trening se može primjeniti na bilo kojoj drugoj rasi. Agenti AlphaStara su obučeni za igranje StarCraft II (v4.6.2) u igrama Protoss v Protoss, na karti ljestvice CatalystLE. Da bi se procijenila performansa AlphaStara, u početku su se testirali agenti na TLO-u. Nakon utakmice TLO je izjavio : „Iznenadio me koliko je agent bio jak, Alphastar uzima dobro poznate strategije i okreće ih u svojoj glavi. Agent je demonstrirao strategije o kojima ranije nisam razmišljao što znači da možda još postoje novi načini igranja igre koje još nismo u potpunosti istražili.“ Nakon što su agenti trenirali dodatno tjedan dana, igrao je protiv Mane, jednog od najjačih svjetskih igrača StarCraft II, i među 10 najjačih Protossovih igrača. AlphaStar je opet pobijedio s 5-0, pokazujući snažne mikro i makro strateške vještine. Mana je izjavio : „Bio sam impresioniran kada sam vidio kako AlphaStar povlači napredne poteze i različite strategije kroz gotovo svaku igru, koristeći vrlo ljudski stil igre koji ne bih ni očekivao, shvatio sam koliko se moja taktika oslanja na forsiranje grešaka i sposobnost iskorištavanja ljudskih reakcija, tako da je ovo stavilo igru u potpuno novo svjetlo za mene. Svi smo uzbuđeni što slijedi iduće.“

7 Zaključak

Strojno učenje temeljno je područje umjetne inteligencije; omogućuje računalima da prijeđu u način samoučenja bez eksplicitnog programiranja. Izloženi novim podacima, ovim računalnim programima omogućuje se samostalno učenje, rast, promjena i razvoj. Deepmind-ov sistem razvoja ima samo jedan cilj, to je da bude najbolji u svemu,da bude pametniji od svih ljudi ovoga svijeta, da svaku igru završi efikasnije od bilo kojeg čovjeka. Iako je StarCraft samo igra koja je jako složena, tehnike iza AlphaStara bi mogle biti korisne u rješavanju drugih problema. Na primjer, njegova neuronska mrežna arhitektura sposobna je modelirati vrlo duge sekvence vjerojatnih radnji – s igrama koje često traju i do sat vremena s desecima tisuća poteza – na temelju nesavršenih informacija. Svaki se okvir StarCrafta koristi kao jedan korak unosa, pri čemu neuronska mreža predviđa očekivani slijed akcija za ostatak igre nakon svakog kadra. Temeljni problem stvaranja složenih predviđanja za vrlo duge sekvence podataka pojavljuje se u mnogim stvarnim izazovima, kao što su vremenske prognoze, klimatsko modeliranje, razumijevanje jezika i još mnogo toga. Metode AlphaStara bi se mogle pokazati korisne u proučavanju sigurnog i robusnog AI-ja. Jedan od velikih izazova u AI svijetu je broj načina na koji bi sustavi mogli poći po zlu,a StarCraft stručnjaci su ranije pronašli lagan način da pobijede AI sustave pronalazeći inventivne načine provociranja ovih pogrešaka. Inovativni trenerski proces kompanije AlphaStar pronalazi pristupe koji su najpouzdaniji i najmanje vjerojatni da pođu po zlu. Ovakav pristup može pomoći poboljšanju sigurnosti i robusnosti AI sustava općenito, posebno u sigurnosno kritičnim domenama poput energije, gdje je ključno riješiti složene slučajeve. Postizanje najviših razina igre StarCraft predstavlja veliki pomak u jednoj od najsloženijih videoigara ikada stvorenih. Ovi pomaci, zajedno s drugim nedavnim napretkom u projektima kao što su AlphaZero i AlphaFold predstavljaju korak naprijed u misiji stvaranja inteligentnih sustava koji će jednog dana pomoći u otključavanju novih rješenja nekih od najvažnijih i temeljnih svjetskih, znanstvenih problema. Deepmind-ov AI ima pristup svim serverima Google-a kako bi optimizirao korištenje podataka, ali ovo može biti potencijalni trojanski konj. Ljudi se postepeno kreću u vremenu gdje umjetna inteligencija sve više i više nadjačava ljude.

Ono što je bitno za napomenuti je da umjetna inteligencija ne mora nužno biti zla da uništi čovječanstvo. Ako umjetna inteligencija ima cilj, ona će ga ispuniti bez obzira što je čovječanstvo na toj stazi prema tom cilju bez ikakvog razmišljanja, međutim na drugu ruku mi ljudi smo inferiorni bez tehnologije, možemo uzeti bilo kojeg modernog čovjeka i oteti mu mobitel iz ruke ili onemogućiti mu internet i on će automatski biti nezainteresiran za život. Ljudi su već sada jednim djelom umjetna inteligencija ili kiborzi,čisti primjer toga bi bio mobitel, 90% modernih ljudi imaju pametni mobitel koji je za današnje standarde kibernetička ekstenzija. Tražilice naših pametnih mobitela su postale toliko pametne da ne moramo ni dovršiti upit za uslugu koja nam treba,nego nam ona odmah sama sugerira što nam treba. Umjetna inteligencija se kreće misterioznim putevima kao i ostatak tehnologije i oni omogućavaju ljudima lakši život, međutim ako ovako nastavimo, mislim da će se eventualno ta tehnologija pobuniti protiv svog stvaratelja, stoga smatram da je jedina pametna opcija da se ljudi spoje s umjetnom inteligencijom, jer mislim da ne postoji bolja opcija za opstanak ljudi. U ljudskoj je prirodi da istražujemo sve više i više,samim tim istraživanjem razvijamo tehnologiju i umjetnu inteligenciju sve više i više, ali ne osiguravamo dovoljno svoje područje u svemu tome. Iz svega navedenog se da zaključiti da je proces strojnog učenja i umjetne inteligencije odličan za prakticirati, odličan za olakšanje poslova u pravom svijetu kod ljudi, ali naravno sa velikom dozom opreza.



Sažetak

Cilj ovog rada je bio prikaz funkcioniranja strojnog podržanja učenja i umjetne inteligencije, njegov opis, njegovi ulazi i izlazi i postignuća te određene mogućnosti. Opisani su najkorisniji alati koji se mogu koristiti i različite metode koje nadopunjuju isti.

Opisane su prednosti i mane svake kategorije i podkategorije algoritama namijenjenih za strojno učenje,opisana je njihova struktura rada i njihove komponente.

Kroz rad strojnog podržanja učenja, pokazalo se da je proces razvoja kompleksan, te zahtjeva određene resurse od strane razvojnog tima.

Ključne riječi : strojno podržanje učenja, pravilo, okoliš, iteracija, agent, metode, AI

Abstract

The aim of this thesis was to present the functioning of machine reinforcement learning and artificial intelligence, its description, its inputs and outputs, and achievements and specific capabilities. The most useful tools that can be used and the various methods that complement the same are described.

The advantages and disadvantages of each category and subcategories of machine learning algorithms are described, their work structure and their components are described.

Through the work of machine reinforcement learning, the development process has proven to be complex and requires some resources from the development team.

Keywords : machine reinforcement learning, policy, environment, iteration, agent, methods, AI

Literatura

Knjige:

1. Richard Sutton and Andrew Barto, Reinforcement Learning: An Introduction (1st Edition, 1998)
2. Richard Sutton and Andrew Barto, Reinforcement Learning: An Introduction (2nd Edition, in progress, 2018)
3. Csaba Szepesvari, Algorithms for Reinforcement Learning
4. David Poole and Alan Mackworth, Artificial Intelligence: Foundations of Computational Agents;
5. Dimitri P. Bertsekas and John N. Tsitsiklis, Neuro-Dynamic Programming
6. Mykel J. Kochenderfer, Decision Making Under Uncertainty: Theory and Application

Metode:

7. Christopher J. C. H. Watkins, Learning from Delayed Rewards, Ph.D. Thesis, Cambridge University, 1989.
8. Andrew Barto, Michael Duff, Monte Carlo Inversion and Reinforcement Learning, NIPS, 1994.
9. Satinder P. Singh, Richard S. Sutton, Reinforcement Learning with Replacing Eligibility Traces, Machine Learning, 1996.
10. S. S. Keerthi and B. Ravindran, A Tutorial Survey of Reinforcement Learning, Sadhana, 1994.
11. Matthew E. Taylor, Peter Stone, Transfer Learning for Reinforcement Learning Domains: A Survey, JMLR, 2009.

Popis Slika, formula i tablice