Hibrid neuronske mreže, genetskog algoritma i reinforcement learninga za igranje top-down shooter igre

Seminarski rad u okviru kursa
Računarska Inteligencija
Matematički fakultet

Đaković Branko, Filip Kristić, Krčmarević Mladen
brankodjakovic08@gmail.com, filip.kristic96@gmail.com
mladenk@twodesperados.com

August 31, 2019

Abstract

U radu će biti prikazana upotreba reinforcement learninga i genetskog algoritma za generisanje težina neuronske mreže koja igra top-down shooter igru "Shrodinger's shooter". Ovaj pristup spada u Neuro-evoluciju[**?**], sa tim što se menjaju samo težine neruonske mreže dok je struktura fiksna. Za pisanje kodova je korišćen programski jezik C++ i biblioteka FANN sa omotačem za C++. Biće dat kratak opis igre kao i dva rešenja koja se razlikuju po načinu reprezentacije ulaza za neuronsku mrežu.

Contents

1 Uvod

"Shrodinger's shooter" je shooter igra koju smo razvili za predmet Razvoj Softvera u kojoj je cilj igrača da što duže preživi nalete protivnika. Igrica koristi 2D fiziku uz 3D grafiku a mapa je kvadratnog oblika i sadrži zidove. Moguće akcije igrača su: idi gore, dole, levo, desno kao i dijagonalno kretanje njihovom kombinacijom, podešavanje pozicije nišana, repetiranje oružja i pucanje. Zbog kompleksnosti su izbačeni itemi poput pancira i različitog oružja. Cilj neuronske mreže[**?**] je da na osnovu trenutne situacije igrača da njegovu sledeću akciju.

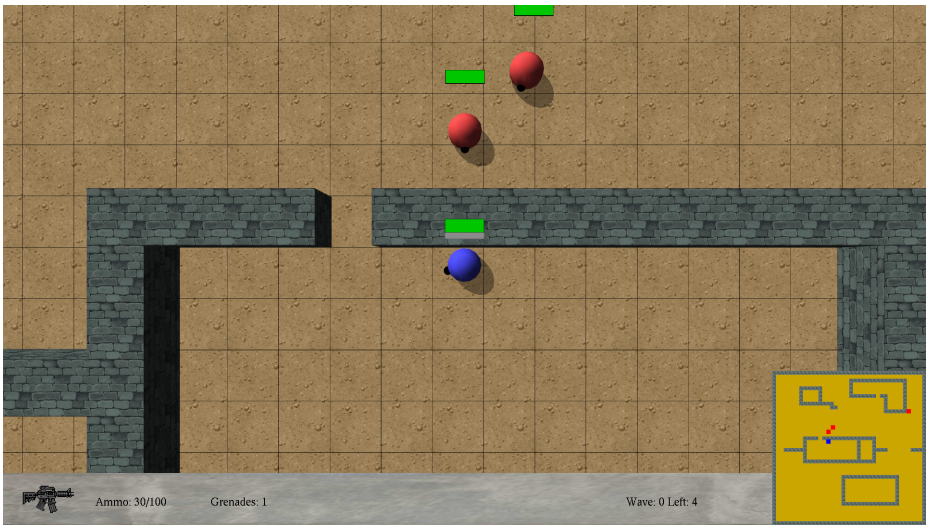

Figure 1: Schrodinger's shooter

2 Opis rešenja

Program ima tri moda izvršavanja:

```
./Schshooter.out
```

koji obuhvata učitavanje rešenja u vidu neuronske mreže i igranje igrice uz grafički prikaz,

```
./Schshooter.out -t
```

koji vrši treniranje bez grafičkog prikaza i

```
./Schshooter.out -tv
```

koji vrši treniranje sa grafičkim prikazom.

Trening započinje kreiranjem početne generacije genetskog algoritma[**?**] (slika 2.[1])kod koje svaki hromozom ima fitness jednak nuli. Hromozomi su predstavljeni sadržajem tj. nizom brojeva u pokretnom zarezu koji predstavlja težine svih veza mreže i jednim brojem u pokretnom zarezu koji predstavlja fitness tog hromozoma. Težine se uzimaju nasumično iz intervala (-10, 10) koji je takođe nasumično izabran. Broj hromozoma u generaciji kao i broj generacija su izabrani uzimajući u obzir veličinu hromozoma te iznose nekoliko stotina u prvom rešenju a nekoliko hiljada u drugom. O ulazu će više biti rečeno u narednom poglavlju.

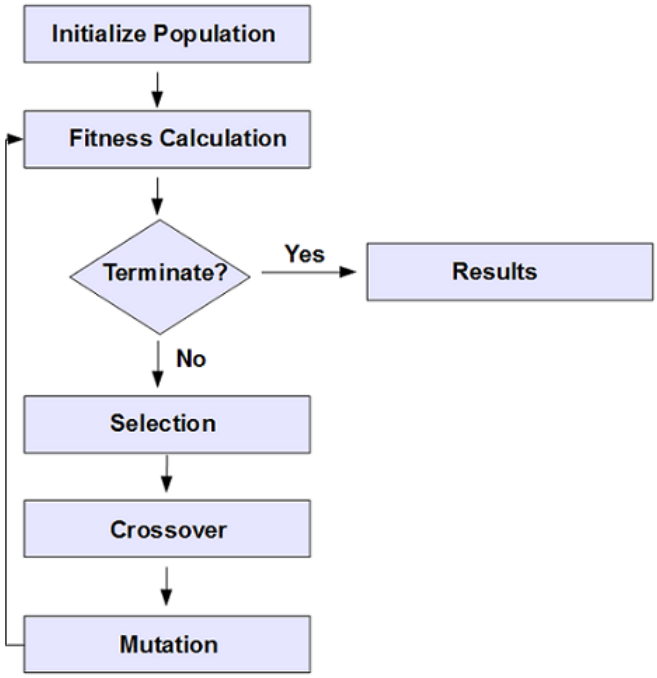

Figure 2: Genetski algoritam

Nakon kreiranja generacije redom se uzimaju hromozomi i težine veza neuronske mreže se postavljaju na sadržaj hromozoma. Veličina ulaznog sloja u prvom rešenju je 191 a 10 u drugom, oba rešenja imaju samo jedan skriven sloj koji je veličine 10 u prvom zbog ogromnog broja veza usled veličine ulaza i 15 u drugom. Za aktivacionu funkciju je korišćna linearna aproksimacija sigmoidne funkcija. Sigmoidna funkcija je oblika[2]

ISPRAVI VELIČINE

[1]Slika preuzeta sa: https://apacheignite.readme.io/docs/genetic-algorithms

[2]Preuzeto iz dokumentacije fann biblioteke. http://leenissen.dk/fann/html/files/fann_cpp-h.html

x - ulaz
y - izlaz
s - nagib i
d - derivacija
raspon: 0 ¡ y ¡ 1

$$y = \frac{1}{1 + e^{-2 \times s \times x}} \quad (1)$$

$$d = 2 \times s \times y \times (1 - y) \quad (2)$$

dok stopa učenja nije podešena pošto nema nikakvog uticaja na dati problem. Izlazni sloj sadrži pet neurona čiji izlazi imaju vrednosti u inetrvalu [0,1] a koji regulišu kretanje gore-dole, levo-desno, treći ugao pod kojim igrač nišani,da li da puca i da li da dopuni municiju. Prva dva izlaza su podeljena u tri intervala [0,0.33], (0.33-0.66] i (0.66-1] koji redom odgovaraju kretanju gore(levo), bez kretanja i dole(desno), treći izlaz se skalira do 360 i ugao igrača se postavlja na datu vrednost a četvrti i peti daju potvrdan odgovor za vrednosti manje od 0.5 a negativan u suprotnom. U svakoj iteraciji programa se ažuriraju pozicija i akcije igrača u zavisnosti od izlaza mreže. Fitness svakog hromozoma se računa na osnovu broja eliminisanih protivnika i količine štete koje su naneli protivnicima po narednoj formuli a najbolje ocenjen hromozom se čuva.

fintess = eliminacije * 50 + šteta * 0.5

2.1 Selekcija, ukrštanje i mutacija

Po obradi svih hromozoma generacije, ukoliko ima još iteracija, vrši se selekcija hromozoma koji će učestvovati u izradi nove generacije. Selekcija se vrši ruletskim pristupom, tako što se računa zbir fitness-a svih hromzoma i svaki ima šansu da bude izabran srazmernu odnosu njegovog i celokupnog fitnessa. Nakon izbora hromozoma koji učestvuju u reprodukciji nasumično se biraju dva roditelja i vrše se ukrštanje i mutacija koji su implementirani na različite načine u prvom i drugom rešenju. Svaki par roditelja kreira dva deteta i to se vrši sve dok ne bude ispunjena nova populacija.

Ukrštanje u prvom rešenju:

i - nasumičan broj od 0 do veličine sadržaja hromozoma;
dete1 = sadržaj roditelja1 do i + sadržaj roditelja2 od i do kraja;
dete2 = sadržaj roditelja2 do i + sadržaj roditelja1 od i do kraja;

Mutacija u prvom rešenju:

t - nasumičan broj u pokretnom zarezu iz intervala [0,1];
Ukoliko je t manje od stope mutacije:
Promeni nasumičan element sadržaja hromozoma;

Ukrštanje u drugom rešenju:

Za svaki element i sadržaja hromozoma uradi:
t - nasumičan broj u pokretnom zarezu iz intervala [0,1];
Ako je t ¡ 0.5:
dete1[i] = roditelj1[i];
dete2[i] = roditelj2[i];
Inače:
dete1[i] = roditelj2[i];
dete2[i] = roditelj1[i];

Mutacija u drugom rešenju:

Za svaki element i sadržaja hromozoma uradi:
t - nasumičan broj u pokretnom zarezu iz intervala [0,1];
Ukoliko je t manje od stope mutacije:
Promeni i-ti element sadržaja hromozoma;

Celokupan navedeni postupak se zatim obavlja dok nije zadovoljen kriterijum zaustavljanja, tj. dok se ne premaši zadati broj iteracija. Najbolja jednika kao i svi hromozomi poslednje generacije se čuvaju u vidu neuronskih mreža u tekstualnim datotekama radi čuvanja progresa i nastavka treniranja.

3 Uporedjivanje rešenja

U ovom poglavlju biće opisana dva eksperimentalna rešenja problema. Njihova suštinska razlika je način na koji je predstavljen ulaz neuralne mreze igrača, tj. način na koji se opisuje trenutno stanje okoline igrača.

3.1 Prvo rešenje:

U prvom rešenju pokušan je pristup predstavljanja celokupne okoline igrača, naime svako polje na ekranu predstavlja jedan ulazni čvor koji moze imati vrednosti: 0 ako je polje prazno, -1 ako se na njemu nalazi protivnički igrac i 1 ako je u pitanju zid. Ovaj unos je poprilično velik jer na ekranu, u datom trenutku, igrač moze da vidi 190 polja, te se to preslikavalo u 190 ulaznih čvorova. Trening je vršen na populaciji od 100 jedinki po generaciji u 250 iteracija.

Karakteristike hardvera na kome je vršen trening:

CPU: intel-i5 2500k
GPU: AMD Radeon 6850
RAM: 4GB DDR3
OS: Ubuntu 16.04

Vreme trajanja je oko 2h posle čega je dobijena konfiguracija koja nije davala praktično dobre rezultate i zaključeno je da ovaj pristup suštinski ne konvergira ka dobrom rešenju te je pokušan drugi pristup sa drastično smanjenjim ulazom neuralne mreze.

3.2 Drugo rešenje:

4 Zaključak