**Vilniaus Užupio gimnazija**

**Oskaras Klimašauskas, Paulius Kurlavičius.**

Objektų skatinamasis mokymas naudojant neuroninio tinklo ir genetinio algoritmo sąsąją.

Kūrybinis/Tiriamasis darbas

Darbo vadovas

Vaidas Raižys

ĮVADAS

Pirmieji neuroniniai tinkai jau buvo sukurti ir sėkmingai apmokyti XX a. viduryje, tačiau jų pritaikymo galimybės buvo labai robota dėl mokymo duomenų stokos ir prastų kompiuterio skaičiavimo pajėgumų. Reikėjo palaukti 60 metų, kad kartu su interneto proveržiu masiniai duomenys būtų lengvai pasiekiami visur – prasideda mašininio mokymo laikai. Dar kiek palaukus ir padobulėjus kompiuterio procesoriams ir vaizo plokštėm, atsirado galimybė tiksliai atkurti realiaus gyvenimo scenarijus kompiuterio ekrane, juose patalpinant objektus sąveikaujančius su aplinka. Taip su kompiuterio pagalba, atliekant didelius skaičiavimus, objektai „išmoksta“ prisitaikyti prie aplinkos pagal mūsų norus.

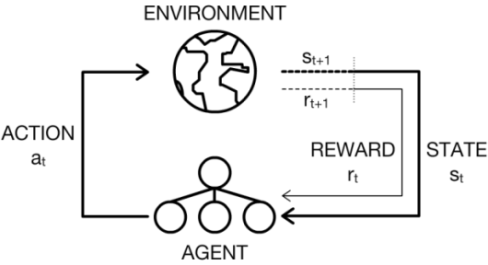
Mes naudodamiesi Unity žaidimų varikliu sukūrėme supaprastintą grafinę aplinką mūsų objektams, kurioje stengtasi simuliuoti kosminę erdvę tarp planetos Žemės ir mėnulio, su tikslu, kad objektai iš pastovios pradinės vietos išmoks pasiekti taikinį, kurio vieta keisis priklausomai nuo žaidėjo įvesties. Tikslą pasiekiamė naudodamiesi daugeliu internetiniu resursų, taip parašydami savo supaprastintą neuroninio tinklo architektūros sudarymo biblioteką ir algoritmus, paremtus skatinamuoju mokymu (angl. *Reinforcement learning*), skirtus objektų apmokymui.

Grafinės aplinkos kūrimas ir skatinamasis mokymas

Saktinamasis mokymas is mašininio mokymo disciplina tyrinėjanti kaip objektai (angl. *Agents* arba *Software agents*) turėtų priimti sprendimus ir judėti sukurtoje aplinkoje, kad gautų kuo didesnės skaitmeninės vertės „apdovanojimą“. Šis mokymo būdas yra vienas iš trijų populiariausių mašininio mokymo būdų kartu su prižiūrimuoju mokymu (angl. *Supervised learning)* ir neprižiūriamuoju mokymu (angl. *Unsupervised learning)*.

**Skatinamasis mokymas susideda iš keturių esminių elementų:**

1. **Objektas.** Programa, kuri yra ”treniruojama“, kad atliktų kūrėjo paskirtą specifinę užduotį
2. **Aplinka.** Pasaulis, realus arba virtualus, kuriame objektas atlieka veiksmus.
3. **Veiksmas.** Veiksmas atlikas objekto, kuris lemia aplinkos būsenos pasikeitima.
4. **Apdovanojimas.** Veiksmo įvertinimas, kuris gali būti teigiamas arba neigiamas.



**Pavyzdžiui:**

**Objektas.** Programa kontroliuojanti robotą.

**Aplinka.** Realus pasaulis.

**Veiksmas.** Vienas iš keturių veiksmų: pirmyn (1), atgal (2), kairėn (3), dešinėn (4).

**Apdovanojimas.** Teigiamas kai robotas sėkmingai pasiekia tikslą. Neigiamas jei jis nukrenta, per ilgai užtrunka, ne ten nuvažiuoja.

Šiame pavyzdyje robotas gali save „išmokyti“ važiuoti efektyviau ir tiksliau reaguojant į savo gautas apdovanojimo vertes.

**Mūsų sukurto proceso aprašymas:**

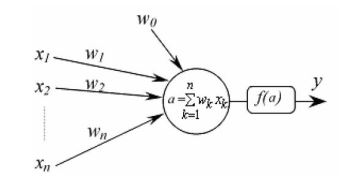
Sukūrėme virtualią aplinka naudodamiesi Unity žaidimų varikliu, kuri susideda iš komponentų: pradinio objektų iniciavimo taško (Žemė ekrano apačioje), tikslo (kita planeta viršuje), kurio vieta priklauso nuo stebėtojo įvesties, objekto (raketos). Simuliacijos pradžioje yra inicijuojama populiacija originalaus objekto kopijų (generacija), sukuriamas ir kiekvienam objektui priskiriamas neuroninis tinklas su randominėmis reikšmėmis. Objektai atlieka veiksmus pagal neuroninių tinklų gautas išvestis, po kiek sekundžių generacija nutraukiama, surušiuojami objektai apdovanojimo vertės tvarka, kuri labiausiai priklauso nuo pasiekto atstumo iki tikslo. Geriausiai pasirodžiusiųjų neuroniniai tinklai yra sumaišomi (angl. *crossover),* pradedama nauja generacija, su viltimi, kad objektai atlikts savo užduotį geriau nei praeitą kartą.

Dirbtiniai neuroniniai tinklai

Dirbtiniai neuroniniai tinklai (DNT) (angl. *artificial neural networks*) – tai informacijos apdorojimo struktūros, netiksliai imituojančios, kai kuriuos gyvųjų organizmų smegenyse vykstančius informacijos apdorojimo procesus. DNT sudaromi iš daugelio tarpusavyje sujungtų labai paprastų skaičiavimo elementų. Šie elementai jungiami vieni su kitais įvairaus stiprumo jungtimis.

**Perceptrono (DNT vieno neurono) architektūra:**

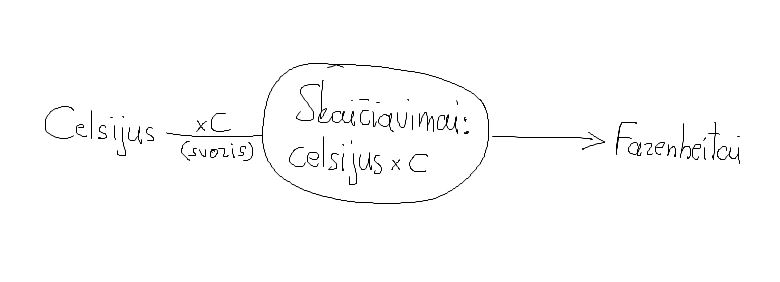
Labiausiai paplitusi ir paprasčiausia dirbtinio neurono schema pavaizduota paveiklėlyje apačioje. Čia *x1, ..., xn* žymi neurono įvestis. Jungčių „stiprumas“ nustatamas pagal svorių turimą vertę (Realieji skaičiai), kurie yra pažymėti *w1,...,wn*. Žymėjimas w0 reiškia slenksčio vertę (angl. *bias,* dažniausiai skirtas apsisaugoti 0 išvesties). Neurono viduje atliekama įvesčių ir svorių sandaugos suma, kuri po to perleidžiama pro aktyvacijos funkciją (pati paprasčiausia yra vadinama *sigmoid,* paverčianti visas vertes atitinkamai nuo 0 iki 1), skirta „sureguliuoti“ gautą rezultatą, o *y* – neurono išvestis.

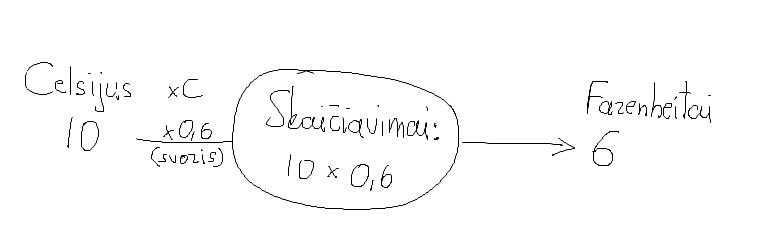


**DNT mokymas:**

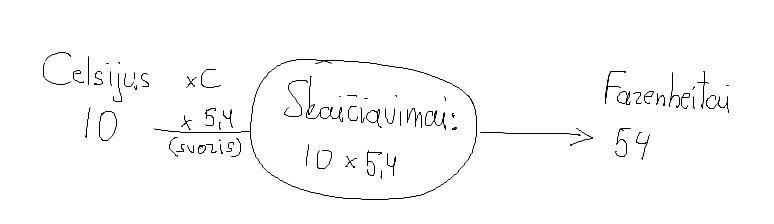
|  |  |  |
| --- | --- | --- |
| Teisingo pvz. nr. | Celsius (Įvestis) | Farenheitai (Išvestis) |
| 1. | 10 | 50 |
| 2. | -5 | 23 |

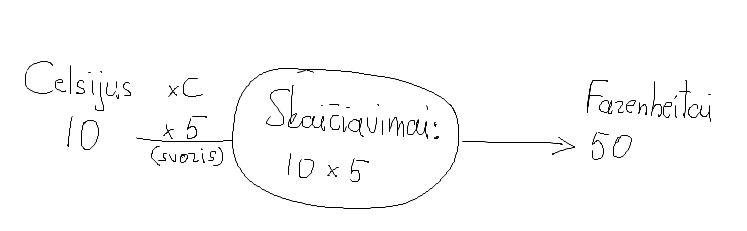
Neuroninis tinklas, inicijuotas su atsitiktinimėmis reikšmėmis duos tokio pačio reikšmingumo išvestį (veikiausiai mums netinkamą), kad tai pakeistume reikia neuroninį tinklą „apmokyti“. Paimkime paprastą pavyzdį (Prižiūrimajo mokymo). Turime lentelę

ir perceptroną su viena įvestimi ir viena išvestimi, todėl turintis tik vieną svorį, žinome tik tai, kad ši problema yra tiesinė, todėl farenheitai = celsijus \* c (c = *const,* bet dar nežinome jos reikšmės):

Kadangi svoriai (šiuo atveju *c*) pradžioje yra inicijuojami su betkokiomis reikšmėmis, tą padarykime ir mes:

Pagal lentelės duomenis matome, kad atsakymas nėra teisingas, tuo labiau, galime pasakyti kiek neteisingas. *50 – 6 = 44;* mūsų paklaida yra 44.

Taigi, ką daryti toliau, mes žinome, kad atsakymas neteisingas ir kiek neteisingas. Reikia bandyti taisyti mūsų svorio reikšmę. Žinodami, kad ši problema yra tiesinė, galime nuspėti, kad padidinus vieną iš daugiklių atsakymas bus didesnis, pažvelgus į lentelę – tokio mums ir reikia. Tačiau kiek reikia didinti svorį? Tai mums pasako mūsų gauta paklaida, kuo ji didesnė – tuo labiau reikia keisti svorį.

Patikrinę rezultatą gauname paklaidą lygią -4 (tikrasis atsakymas - gautas atsakymas). Paklaida daug mažesnė, tačiau šį kartą ji yra neigiama – rezultatą ”peršokom“, todėl reikės svorį mažinti, o pažvelgus į paklaidos vertę, jį reikia koreguoti labai nedaug, todėl pabandome paskutinį kartą:

Vėl rezultatą patikrinus paklaida tampa lygi nuliui – tinklas išmokytas, atsakymas teisingas! Dabar mūsų gautą svorį galima išsaugoti ir naudoti kitiems skaičiavimams.

Žinoma šis variantas yra labai stipriai supaprastintas, bet tinklo mokymo esmė išlieka ta pati – rasti optimalius svorius. Tam yra kuriami sudėtingi algoritmai pvz.: vieni yra skirti susieti gautą paklaidą ir ją paskirstyti atitinkamai per kelis ar keliasdešimt svorių, atsižvelgiant į tai kiek jie prie tos paklaidos prisidėjo (angl. *backpropagation)*, kiti mutuoja svorius ir maišo juos su kitų neuroninių tinklų svoriais, vėlgi atsižvelgiant į paklaidą (Skatinamasis mokymas).

**Kaip veikia mūsų algoritmas:**

Genetinis algoritmas - algoritmas, paremtas gyvybės bruožų pokyčiais per kartas (Evoliucija). Genetinis algoritmas pasiremia evoliucija trimis būdais.

1. Gamtinė atranka – labiausiai prie aplinkos organizmai išgyvena.
2. Persikryžiavimas - genų mainai tarp tėvų gimdant naują palikuonį.
3. Mutacija – palikuonio genetinės medžiagos pokytis.

Simuliuoti gamtiniai atrankai programoje naudojamos skaitinės „prisitaikymo“ (angl. fitness). Kiekvienas objektas populiacijoj atsiranda su atsitiktinai išrinktais genais ir bando reaguoti į aplinką. Kuo arčiau objektas yra tikslo tuo labiau jo „prisitaikymas“ didėja. Kiekvieną kartą objektam atliekant veiksmus aplinkoje ir kartai pasibaigus, algoritmas juos surūšiuoja pagal jų „prisitaikymo“ skaitines vertes. Tie, kurie pasirodė geriausiai tarpusavyje palieka palikuonius persikryžiavimo būdu kitai kartai.

Persikryžiavimo metu objektas A ir objektas B sukuria naujų palikuonių. Tai daroma maišant jų abiejų genus gaunant 2 naujus palikuonius.

Mutacija - naujų palikuonių genų atsitiktinis pakitimas. Mutacijos paskirtis yra padėti išvengti prisitaikymo funkcijos vietinio minimumo (angl. local minimum). Be mutacijos vienintelės genų galimos vertės būtų tos, kurias gauna pirmoje kartoje tik sumaišytos naujose kartose. Mutacija leistų potencialiai išbandyti kiekvieną galimą vertę, o radus labiau tinkančią vertę užduočiai objektas gebės surinkt daugiau „prisitaikymo taškų“ ir perduos savo genus kitom kartom. Absoliutus minimumas (angl. Global minimum) - visos vertės yra optimalios užduočiai atlikti.

Neuroniniam tinkle gyvybę sudarantys genai yra pačio tinklo sudarantys elementai: svoriai, aktyvacijos funkcija, sluoksniai, neuronu kiekis. Genetiniui algoritmui privaloma yra svoriai, kitu elementu modifikavimas yra pasirinktinas.

„Prisitaikymo taškai“ labiausiai priklauso nuo atstumo tarp objekto ir tikslo, jį pasiekus, objekto „Prisitaikymo taškai“ padidinami 10 kartų, kad užtikrintų, kad objektas paliks palikuonių kitai kartai. Visiems objektams pabandžius skristi naudojamas Šelo rikiavimo algoritmas, kad greit surūšiuotų raketas pagal jų prisitaikymą. Vėliau naudojamas persikryžiavimas, kur naudoja geriausiai pasirodžiusių 50% raketų ir sumaišo jų genus (svorius) tarpusavyje, kad gautų naujus palikuoniu kitai kartai. Kiekvienas naujai sukurto palikuonio svoris gali atsitiktinai mutuoti, tam, kad toliau ieškotų efektyvesnio būdo spręsti užduočiai.

Iš objekto yra brėžiamos 7 linijos kas 30°, kai linija atsitrenkia į kita objektą (planetą) tas atstumas tarp raketos ir atsitrenkto taško tampa neuroninio tinklo įvesta informacija.

**Rezultatai/Išvados**

Norimo tikslo pilnai įgyvendinti napvyko, objektai neišmoksta sekti objekto visais įvesties atvejais. Tačiau objektai išmoksta pasiekti tikslą, kurio koordinatės keičiasi labai minimaliai. Tai reiškia klaidą algoritme, bet dėl mūsų ribotų žinių jos rasti nepavyko.