Rīgas 10.vidusskola

Datu apjoma ietekmēšana uz mašīnmācīšanās rezultātiem «Snake» spēles piemērā

Zinātniskās pētniecības darbs informātikas sekcijā

Darba autors:
Jegors Baļzins

Darba vadītājs:
informātikas skolotājs
Vsevolods Antjufejevs

Rīga, 2018.

Anotācija

Datu apjoma ietekmēšana uz mašīnmācīšanās rezultātiem «Snake» spēles piemērā, Jegors Baļzins, darba vadītājs informātikas skolotājs Vsevolods Antjufejevs.

Darba mērķis: noteikt mašīnmācīšanu rezultātu atkarību no datu apjoma, apmācot programmu spēlet spēli «Snake».

Darba uzdevumi:

- izpētīt teoriju par temātu;
- izveidot vidi programmas apmācīšanai;
- izveidot apmācīšanas programmu;
- apmācīt programmu un notestēt to;
- savākt datus par apmācīšanas un testēšanās procesu;
- izveidot secinājumus.

Darba gaita: izpētīt teoretiska daļa, saņemti un apstrādīti dati par apmācīšanu

Darba rezultāts: mašīnmācīšanās rezultāts aug ar datu apjomu tikai līdz kādam momentam, un augot datiem pēc tā momentā rezultāts jau neaug vai samazinās.

Annotation

Influence of amoun of data on machine learning results on example of «Snake» game, Jegors Balzins, supervisor teacher of Informatics Vsevolods Antjufejevs.

Goal of work: find dependence of machine learning results from amount of data, by teaching the programm to play game «Snake».

Tasks of work:

- explore the the theory on the theme;
- create environment for program to train;
- create training program;
- train program and test it;
- collect training and tasting data;
- make conclusions.

Saturs

Ievads

Mašīnmācīšanās ir populāra mākslīgas intelekta datorzinātņu nozāres apakšnozāre, kas mūsdien ļoti ātri attīstās. Visbiežāk mašīnmācīšanās izmanto tādu uzdevumu risināšanai, kuriem nav precīza vai pietiekami ātra atrisināšanās algoritma. Apmācīšanās procesā programma izmanto lielu datu apjomu, un ir interesānti uzzināt kā datu apjoms ietekmē apmācīšanās rezultātu. Lai to izpētīt labāk apskatīt vieglo piemēru, lai mazāk faktoru ietekmētu rezultatu. Laba ideja ir izpētīt ši jautājumu, apmācot programmu spēlet «Snake» spēli, jo dati ģenerējas spēles processā, un ir viegli saprast kuri parametri ir vissvarigākajie.

Darba mērķis: noteikt mašīnmācīšanu rezultātu atkarību no datu apjoma, apmācot programmu spēlet «Snake».

Darba uzdevumi:

- izpētīt teoriju par temātu;
- izveidot vidi programmas apmācīšanai;
- izveidot apmācīšanas programmu;
- apmācīt programmu un notestēt to;
- savākt datus par apmācīšanas un testēšanās procesu;
- izveidot secinājumus.

Hipoteze: lielākais datu apjoms novedīs pie labāka rezultāta.

1. Teoretiska daļa

1.1. Spēle «Snake»

Spēle «Snake» ir populāra datorspēle izdomāta 1977.gadā, kur spēletājs kontrole maigo būtni, kura izskatās lidzīgi čūskai. Spēles noteikumi ir dažādi dažos versijas, bet koncepcē ir kāda izmēra laukā, kur pārvietojas būtne kaut kadā virzienā, kuru var mainīt spēletais. Būtne sastāv no «galvas» un «astes». Kad galva satiek «barjeru», spēle beidzas, citādi turpinās. Ja būtne apēd «ābolu» aste pagarinās.

1.2. Mašīnmācīšanās

Mašīnmācīšanās ir mākslīgā intelekta apakšnozare, kurā nodarbojas ar tādu algoritmu izstrādi, kuri ļauj datoriem uzlabot lēmumu pieņemšanu, pamatojoties uz empīriskiem datiem. Izmantojot mašīnmācīšanās tehnikas, programma apmācas dot vairā vai mazāk lidzīgo labakājam atbildēm atbildi.

1.3. Pastiprinātas mācīšanās

Pastiprinātas mācīšanās ir mašīnmācīšanās veids, kuras apmacīšanas laikā aģents iedarbojas ar kādu vidi un saņemt pastiprinājuma signālu jeb vides reākciju. Aģents ir programma, prognozē darbību dotājos vides apstākļos, un pēc saņemtajiem datiem par vides reākciju korektē savu prognazēšanas sistēmu.

1.4. «Q-learning»

«Q-learning» ir pastiprinātas mācīšanās tehnika, kad aģents veido noderības tabulu, kurā katrai vides stāvokļa un iespējamo darbību ši stavokļa paras vertībai tiek piešķirta noderība jeb cik noderīgi ir izpildīt ši darbību šajā stavoklī. «Q-learning» algoritms ir ļoti viegls, tāpēc to ir viegli uzrakstīt ar jebkādu populāro programmēšanas valodu. Vispārināta veida «Q-learning» algoritms sastāv no šādiem soļiem:

1. Izveidot tabulu Q un piešķirt visiem $Q_{s,a}$ (kur s ir vides stāvoklis, bet a ir tām derīga darbība) ar nejaušam vertībam kādā diapozonā (tas ir atkarīgs no citiem parametriem)
2. Kaut kādu laiku atkārtot procesu(jo vairāk atkārtot, jo atkartos katrais stavoklis):
3. Saņemt vides stavokli - s un izvelēt darbību ar lielāko noderību no visiem Q_s vertībām un izdarit to.
4. Saņemt balvu - r no vides
5. Atjaunot $Q_{s,a}$ pēc formulas(kur α ir macīšanas temps un γ ir diskonta faktors):

$$Q_{s,a} = (1 - \alpha) + \alpha(r + \gamma \cdot \max Q_{s+1}) \quad (1.)$$

1.5. «Python» programmēšanas valoda

«Python» ir populāra programmēšanas valoda , kuru plaš izmanto mašīnmācīšanās to vieglas sintakses un lielo bibliotēku ar atverto kodu skaitu deļ.

2. Praktiska daļa

2.1. Vides sagatavošana

Izmantojot teoretiskas zinās par spēli «Snake» un programmēšanas valodu «Python», varam uzrakstīt makslīgo vidi.

2.1.1. Palīgkods

Lai vidi un velāk aģentu būtu vieglāk testēt un rakstīt, vispirms uzrakstīsim paligkodu, kur:

1. importēsim palidzīgas bibliotēkas;
2. uzrakstīsim funkcijas kuri veic matematiskas aprēķināšanas, kā distances vai virziena pagrizienu aprēķināšana;
3. uzrakstīsim vides stavokļa izvadīšanas uz ekrāna funkcijas, lai pēc tām būtu vieglāk testēt

2.1.2. «Čūskas» kods

Aprakstīsim «čūsku» ka klasi, kurām būs dati un metodi:

1. «čūska» ka koordinātu saraksts;
2. kustības virziens;
3. vai apēdis «ābolu» nākamāja soli;
4. funkcija, kura pagriez virzienu pa labi;
5. funkcija, kura pagriez virzienu pa kreisi;
6. funkcija, kura teic «čūskai» izdarit soli virzienā
7. palīgfunkcijas

2.1.3. Spēles kods

Spēlei izveidosim funkcijus, lai no viena koda būtu iespēja dažreiz to spēlēt. Inicializēsim lauku kā 2D masīvu 20x20, kur sienas nav brīvas, un visas vertības un sāksim bezgalīgu ciklu. Katra iteracija aģents vai spēlētais izveles darbību, un «čūska» to izdarīs, ja «čūska» satiks barjeru vai savu asti, tad spēle beigsies, ja «čūska» satiks «ābolu», tad «čūska» palielinās uz 1, un «ābols» parādīsies kāda neaizņmta kvadrātā laukā, atjaunosim visas vertības un atkartosim.

2.2. Aģenta gatavošana

Uzrakstisim aģentu kā klasu kas satur noderības tabulu un izmanto, atjauno un saglāb ta datus. Aģents atjaunos dauts tabulā pēc 1. formulas (sk. 1.3. «Q-learning»).

2.3. Dati par vides stāvokli

Mēs zinām, ka aģents izmanto informāciju par vides stāvokli, bet kādu informāciju mūms vajag dot? Vajag saprast, ka jo vairāk daždo stāvokļu var būt, jo labākus atbildes dos aģents, bet vairāk laiku vajadzēs apmācīšanai. Mūsu uzdevums nav dabū labāku rezultātu, bet noteikt kā noteikt rezultātu atkarību no datu apjoma. Tāpēc izmantosim tikai informāciju relatīvu(skatoties «čūskas» kustības virzienā) «ābola» pozīciju un kvadrātus priekšā, pa labi un pa kreisi no «čūskas» galvas - vai nomirs ja kustoties uz viņiem, kas dos apmēram 10000 dazādo vides stāvokļu.

2.4. Aģenta trenēšana

Lai notrenēt aģentu definēsim cik iterāciju jāizdara trenēšanā laikā. Lai negaidīt daudz optimāls iterāciju skaits būs 100000, pēc tādu meginājumu skaitu tām jau jārada kaut kādu rezultātu. Arī definēsim macīšanas tempu, diskonta faktoru un balvas par dažadiem sasniegumiem. Kamēr nebūs laba rezultāta. Kad to sasniegsim, trenēsim aģentu dažreiz ar dažiem iteraciju skaitiem. Jo vairā ir iterāciju, jo vairāk datu izmanto aģents, tāpēc varam uzskatīt, ka skatoties uz rezultāta no iterācijas jeb soļu skaitu, skatamies uz rezultāta no datu apjoma atkarību. Noderības tabulus saglābāsim failā.

2.5. Aģenta testēšana

Lai notestēt aģentu atjaunosim tabulu no saglābāta faila un nospelēsim dažas spēles, skatoties, vai «čūskas» neit kāda ciklā, ja iet, tad sākt jauno. Rezultātus saglābsim.

2.5. Dati par aģenta trenēšanu un testēšanu

Izdarot šo procesu, es izvelejos $\alpha = 0.3$; $\gamma = 0.3$; dot balvu -10 par pārvietošnos talāk no «abola», -100 par spēles beigšānu, +100 par pārvietošnos tuvāk «ābolām», +500 par «ābola» āpēšanu. Staitistika par aģenta trenēšanu sk. tabula 1.

2.6. Datu apstarāde

Vispirms, vajag teikt ka trenēšanas dati veidojas nejauši tāpēc trenēšanās procesā var radīties anomāles, kā 5. un 9. mēģinājumā (sk. 1. tabulu). Palielinot soļu skaitu līdz 1000000 rezultāts aug, neproporcionāli, bet aug, bet līdz 10000000 smazinās(sk. 1. tabulu). Tās var bū saistīts ar to, ka ir lielāka varbūtība tikt anamāli, bet ja to tikt, tā neļauj augt rezultātam pēc tā. Nav iespējas pārskātit visus soļus, lai atrāst to, bet ir varam pieņiemt, ka «čūska» sāk staigāt cikliski un nevar beigt.

Secinājumi

Darba mērķis bija sasniegts, mašīnmācīšanu rezultāta atkarība no datu apjoma, apmācot programmu spēlet «Snake», bija atrāsta.

Darba uzdevumi ir izpildīti, teoretiska daļa bija izpētīta, vide izveidota, programmabija apmacīta, dati par apmācību bija saņemti un apstradāti, secinājums par datiem ir izveidots.

Dati paradīja, ka hipoteze bija nepareiza, un pēc tiem varam secināt, ka mašīnmācīšanās rezultāts aug ar datu apjomu tikai līdz kādam momentam, un augot datiem pēc tā momentā rezultāts jau neaug vai samazinās.

Ši darbs būs piejams visiem internetā (sk. 1. pielikumu) , lai izmantot to pētniecība vai apmacībā.

Literatūras saraksts

- Себастьян Рашка «Python и машинное обучение.» ISBN 978-5-97060-409-0
- Саттон Ричард С., Барто Эндрю Г. «Обучение с подкреплением» ISBN 978-5-94774-351-7

Materiāli no internetā

- https://docs.python.org/3/
- http://cs229.stanford.edu/proj2016spr/report/060.pdf
- http://people.revoledu.com/kardi/tutorial/ReinforcementLearning/index.html

Pielikumi

tabula 1.

mēģinājums	vidēja punktu skaits apmācībā	lielākais punktu skaits apmācībā	vidējais soļu skaits dzīvē apmācībā	vidēja punktu skaits testēšanā	lielākais punktu skaits testēšanā	vidējais soļu skaits testēšanā
1 - 10000 soļi	3.75	10	85.48	4.3	11	422
2 - 10000 soļi	3.77	12	64.94	4.34	11	405.84
3 - 10000 soļi	3.89	19	65.8	4.76	11	302.37
4 - 100000 soļi	7.14	25	96.71	9.44	29	129.27
5 - 100000 soļi	4.10	11	645.17	4.78	15	488.53
6 - 100000 soļi	6.95	23	95.69	8.65	30	116.17
7 - 1000000 soļi	13.74	42	188.96	18.87	34	261.39
8 - 1000000 soļi	13.53	44	190.26	17.85	34	255.28
9 - 1000000 soļi	5.5	20	2092.05	6.34	18	250.21
10 - 10000000 soļi	5.68	22	18450.18	7.56	19	134.93
11 - 10000000 soļi	7.15	29	9337.07	8.93	37	241.16
12 - 10000000 soļi	6.82	23	10752.69	9.4	23	129.97

1. pielikums
Visi dati izveidoti pētijuma gaita ir pieejami:
https://github.com/vebgor/QLearningSnake