Raport Assignment 5
Flappy Bird using Q learning

Membri echipei: Repede Monica-Gabriela, Murgulet Andrei

Acest raport are ca scop documentarea experimentelor pe care le-am facut pentru a realiza un agent inteligent pentru jocul Flappy Bird, cat mai performant, utilizand Q-learning.

Modelul 1 (11,1)

Arhitectura modelului

Reteaua neuronala folosita pentru aproximarea valorilor Q are urmatoarea structura:
input: imagini preprocesate 66*66*4 (4 imagini consecutive de 66*66)
-Stratul 1: convolutional cu 16 filtre de 8x8(kernel), stride de 4 si functia de activare Relu
-Stratul 2: convolutional cu 32 filtre de 4x4, Stride 2 si functia de activare Relu
-Stratul 3: Fully connected, out: 256 de neuroni si functia de activare Relu
-Stratul 4: De iesire, strat complet conectat liniar, cu o singura iesire pentru fiecare actiune (2 actiuni, 0 si 1)
-> aceasta arhitectura a retelei este preluata din lucrarea “Playing Atari with Deep Reinforcement Learning”- link document
Prepocesarea imaginilor a constat din
-resize-ul acestora in (84, 110) urmat de un crop a imaginii obtinute in dimensiunea (66,66)-> se elimina pamantul, spatiul liber de dinaintea pasarii si se limiteaza vederea urmatoarelor pipe-uri
-conversie in tonuri de gri(COLOR_BGR2GRAY) pentru eficienta.

Hyperparametri

- episodes = 100 000 - numarul de episoade pe care se antreneaza agentul
- experience_memory = 50000 - dimensiunea pentru Experience Replay pentru a stoca cele mai recente frame-uri
- target_update = 10000 numarul de pasi intre actualizarile retelei
- update_freq = 4 -numarul de pasi intre actualizarea modelului Q-network
- frame_skip = 3 - atunci cand alegem sa sarim(1), se da skip la 3 frameuri (si se alege 0- adica nu face nimic)
- epsilon ia valori intre 1 si 0.1 liniar timp de 1 milion de pasi(alegeri de 0 sau 1) si apoi mentinut la 0.1
- RMSProp cu mini batchuri de 32

Sistemul de recompense: pe langa reward-ul din joc propriu-zis, urmarim unele scenarii:

1. pasarea se loveste in primul pipe(tinde sa faca salturi repetate pentru a nu se lovi de pamant si a primi o recompensa mai mare; la primul pipe ajunge la 101)->pentru a evita acest comportament de supravietuire neindicat, o penalizam cu -100
2. daca trece de 101 crestem cu +10

Rezultate obtinute:

Pentru aceste date pe train in 100 000 de episoade rewardul maxim este de 1519 ceea ce in pipe-uri trecute reprezinta 38.

Pe test obtinem rezultate mai facile maximul fiind de 64 de pipe-uri trecute deoarece eliminam elementul de alegere random care persista in continuare pe train(prob 10%). Media insa este in jur de 25 de pipe uri (la 20 de episoade)

Modelul 2 (3,1)
Acelasi model, insa **hyperparametrii** au suferit usoare modificari:

target_update= 1000
politica de actualizare a epsilonului: pentru inceput am pornit cu modificare liniara pentru 100 000 de pasi, apoi cand a ajuns la episodul 38 000, pasul 516613, l-am modificat pentru 1 milion de pasi (epsilon vechi=0.1 , epsilon nou=0.54)

Rezultate obtinute:

Pentru aceste date pe train in 100 000 de episoade rewardul maxim este de 1690 ceea ce in pipe-uri trecute reprezinta 43.

Pe test obtinem rezultate mai facile maximul fiind de 113 pipe-uri trecute. Media insa este tot in jur de 30 de pipe uri (la 20 de episoade)

! De mentionat este ca in ambele cazuri rularea a mai fost oprita si astfel s-a pierdut memoria din bufferul de experience replay, retinandu-se in memorie doar modelul DQN pentru cand s a reluat rularea, iar de aici sunt posibile discrepante in continuitatea scorului !

Modelul 3 (12,1)

Arhitectura modelului

Reteaua neuronala folosita pentru aproximarea valorilor Q are urmatoarea structura:
-> link paper
input: imagini preprocesate 110*84*4 (4 imagini consecutive de 110*84)
-Stratul 1: convolutional cu 32 filtre de 8x8(kernel), stride de 4 si functia de activare Relu
-Stratul 2: convolutional cu 64 filtre de 4x4, Stride 2 si functia de activare Relu
-Stratul 3: convolutional cu 64 filtre de 3x3, Stride 1 si functia de activare Relu
-Stratul 4 ascuns: Fully connected, out: 512 de neuroni si functia de activare Relu
-Stratul 5: De iesire, strat complet conectat liniar, cu o singura iesire pentru fiecare actiune (2 actiuni, 0 si 1)

Prepocesarea imaginilor a constat din
1. crop-ul imaginii - se elimina pamantul, spatiul liber de dinaintea pasarii, iar in imagine apare doar un pipe; al 2 lea pipe apare abia cand pasarea este intr-un pipe, aproape de final
2. resize-ul acestora in (84, 110)
3.conversie in tonuri de gri(COLOR_BGR2GRAY) pentru eficienta

Hyperparametri

- episodes = 11 000 - numarul de episoade pe care se antreneaza agentul
- experience_memory = 50000 - dimensiunea pentru Experience Replay pentru a stoca cele mai recente frame-uri
- target_update = 10000 numarul de pasi intre actualizarile retelei
- update_freq = 4 -numarul de pasi intre actualizarea modelului Q-network
- frame_skip = 3 - atunci cand alegem sa sarim(1), se da skip la 3 frameuri (si se alege 0- adica nu face nimic)
- epsilon ia valori intre 1 si 0.0001 liniar timp de 1 milion de pasi(alegeri de 0 sau 1) si apoi mentinut la 0.0001
- RMSProp cu mini batchuri de 64

Sistemul de recompense: pe langa reward-ul din joc propriu-zis, urmarim unele scenarii:

1.pasarea se loveste in primul pipe(tinde sa faca salturi repetate pentru a nu se lovi de pamant si a primi o recompensa mai mare; la primul pipe ajunge la 101)->pentru a evita acest comportament de supravietuire neindicat, o penalizam cu -50
2.daca trece de 101, insa moare, penalizam cu -10
3. daca nu a murit si a trecut de 101, adunam +1
4.daca nu a murit insa nu a trecut de primul pipe (101) adunam +0.1

Rezultate obtinute:

Pentru aceste date pe train in 11 000 de episoade rewardul maxim este de 8720 ceea ce in pipe-uri trecute reprezinta 233.

Pe test obtinem rezultate mai facile maximul fiind de 387 pipe-uri trecute.

De asemenea am rulat de 6 ori cate 20 de episoade pentru a observa performanta agentului si am trecut scorurile (in functie de numarul de pipe-uri) in tabelul de mai jos.

	A	B	C	D	E	F	G	H	I
	Tabelul_1								
1	# Nr_ep	# Incercare 1	# Incercare 2	Incercare 3	Incercare 4	Incercare 5	Incercare 6		
2	1	44	104	49	47	23	43		
3	2	71	59	21	29	3	23		
4	3	34	117	6	31	201	39		
5	4	94	10	141	1	4	12		
6	5	53	3	9	72	9	11		
7	6	77	79	59	40	37	9		
8	7	54	3	44	69	67	3		
9	8	27	31	17	8	3	97		Average
10	9	25	141	111	29	91	121		49,55
11	10	73	25	149	93	5	109		
12	11	19	111	89	44	27	49		
13	12	16	28	77	2	57	56		
14	13	19	19	6	49	42	55		
15	14	4	18	105	100	89	9		
16	15	27	161	2	27	26	13		
17	16	3	4	37	237	149	40		
18	17	44	17	20	2	19	171		
19	18	116	9	107	11	48	20		
20	19	5	13	70	13	76	19		
21	20	23	129	19	7	24	89		
22		41,4	54,05	56,9	45,55	50	49,4		

Dupa cum se poate observa, modelul din urma a produs cele mai bune rezultate atat pe train cat si pe test; acesta este modelul la care am ramas.