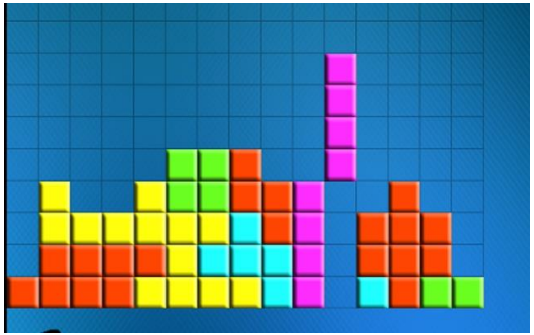
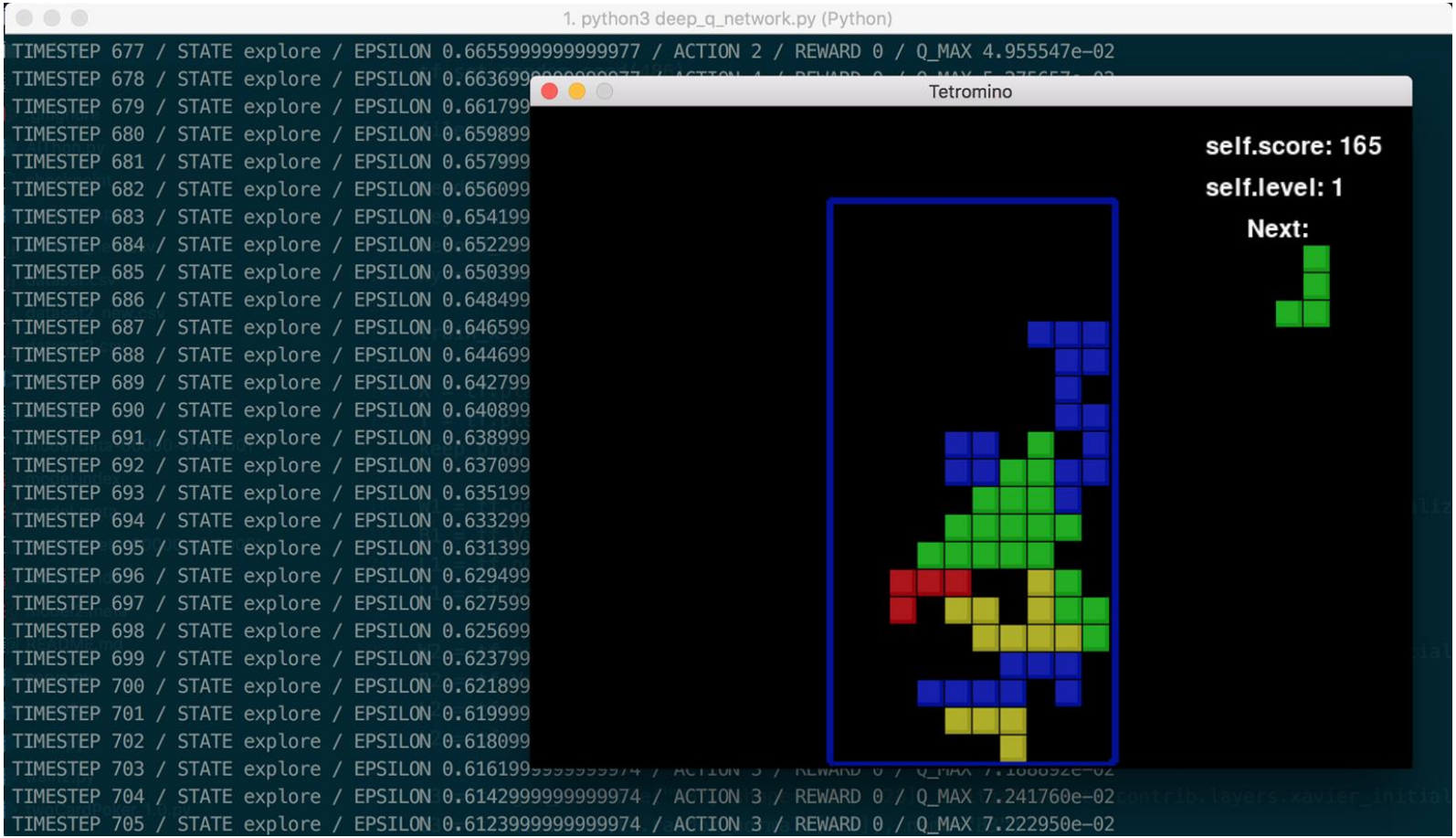
OpenAIThon

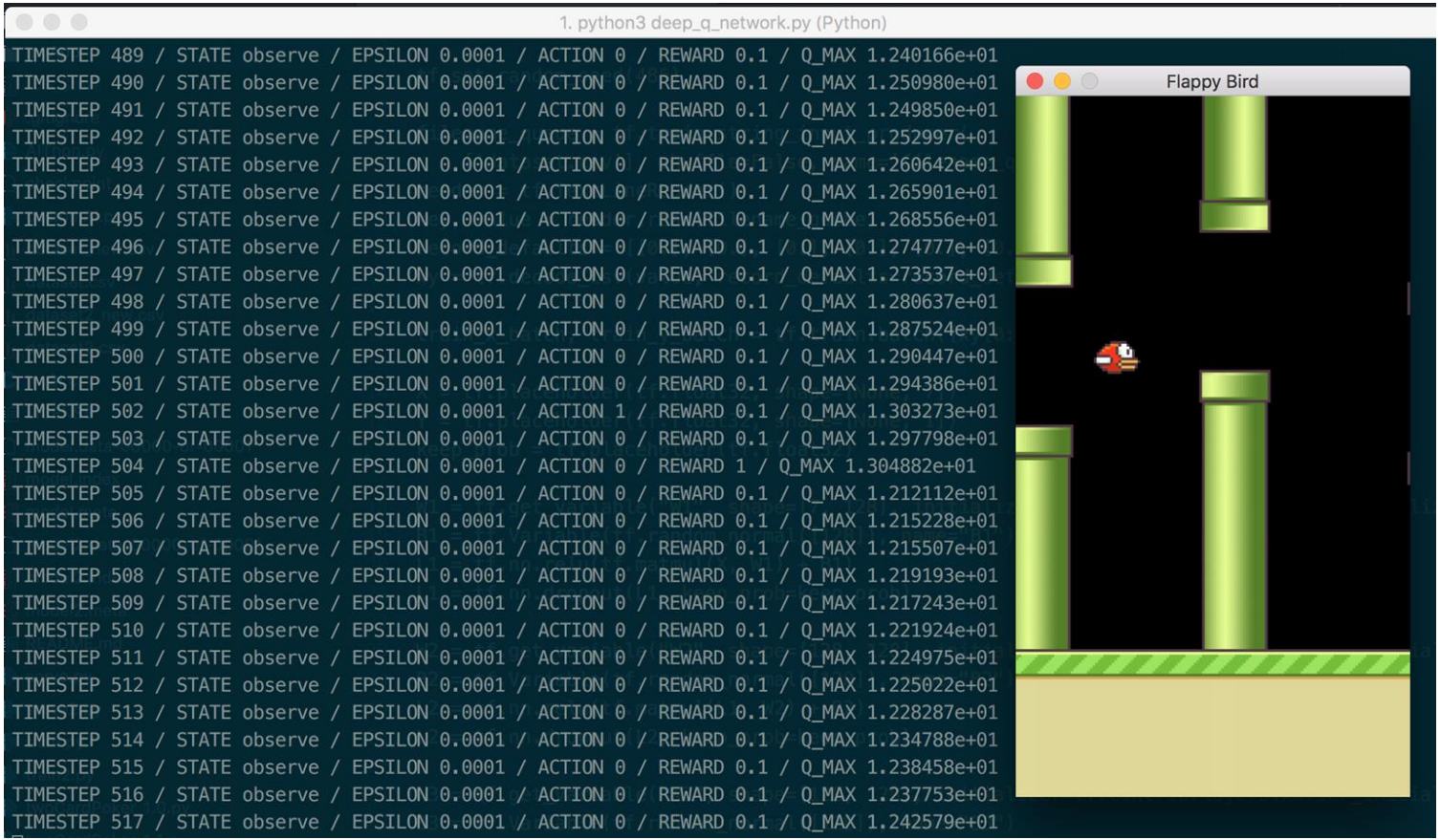
일시 : 17-09-17
장소 : 서울시립대 강촌수련원
팀장 : 최순호
팀원
- 김장현
- 임동건
- 송남주
- 류강현

Motivation





https://github.com/asrivat1/DeepLearningVideoGames



https://github.com/yenchenlin/DeepLearningFlappyBird

우리도 이런 Game AI를 만들어 보자!

Used Open Source

Python

TensorFlow

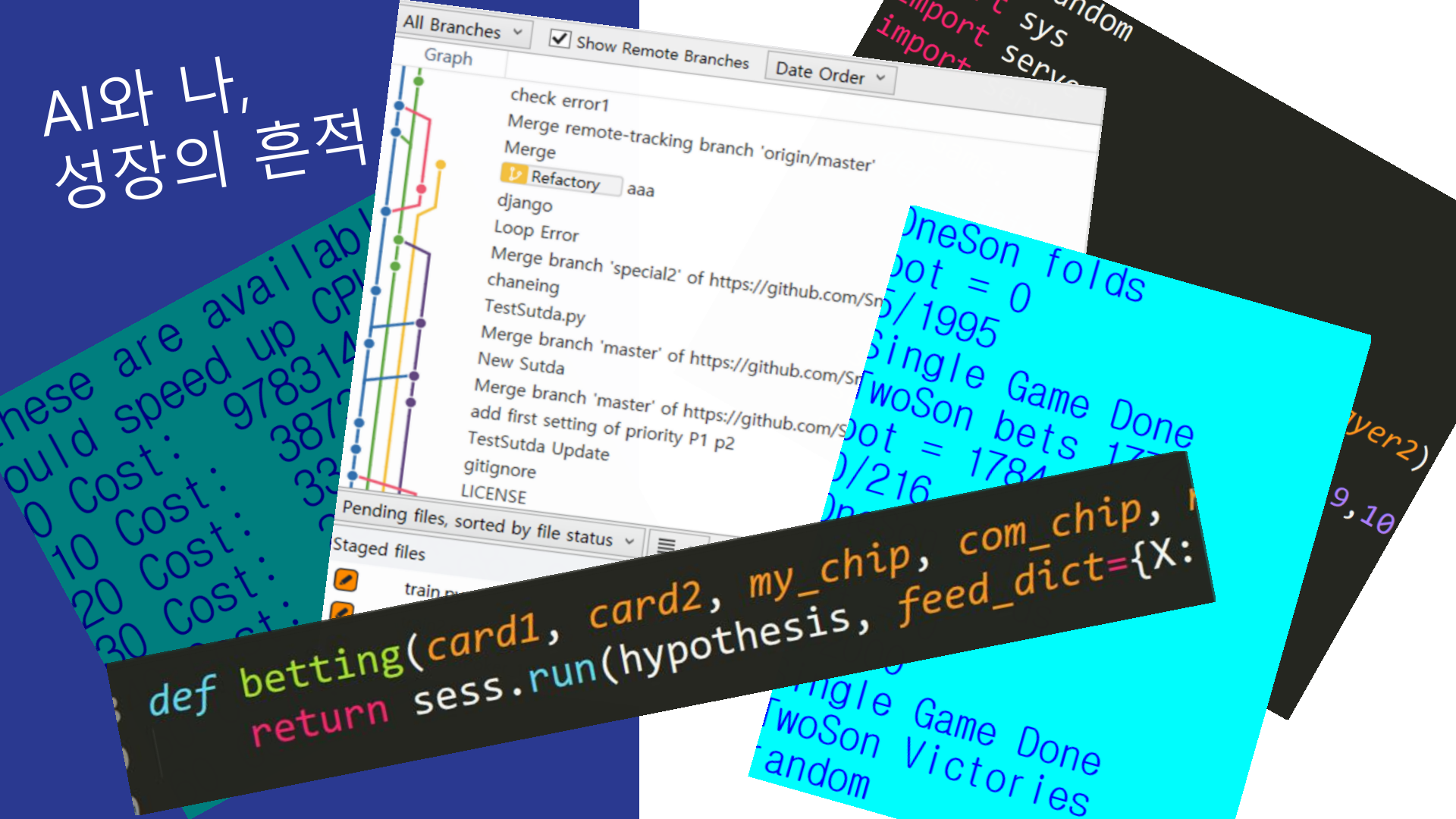
Git

Model Info

- 나의 카드 1, 나의 카드 2, 남은 자신의 칩, 남은 상대방의 칩, 나의 베팅 칩 수, 상대의 베팅 칩 수, 베팅 횟수
- 이번 베팅에서 몇 개의 칩을 베팅할 것인지 결정
- batch_size=1000, layer=4. shape=128, step=5001
- 초기값 설정 xavier_initializer
- 활성화 함수 ReLU
- AdamOptimizer

그냥 좋다는 건 다 썼다.

AI와 나, 성장의 흔적


All Branches
Show Remote Branches
Date Order
Graph
check error1
Merge remote-tracking branch 'origin/master'
Merge
Refactory
aaa
django
Loop Error
Merge branch 'special2' of https://github.com/Sn
chaneing
TestSutda.py
Merge branch 'master' of https://github.com/Sn
New Sutda
Merge branch 'master' of https://github.com/S
add first setting of priority P1 p2
TestSutda Update
gitignore
LICENSE
Pending files, sorted by file status
Staged files
def betting(card1, card2, my_chip, com_chip,
return sess.run(hypothesis, feed_dict={X:
import sys
import
OneSon folds
ot = 0
5/1995
Single Game Done
TwoSon bets
ot = 1784
0/216
ngle Game Done
TwoSon Victories
andom
yer2)
9,10

Step 1.

랜덤 베팅을 하는 컴퓨터 대결로 만든 데이터 셋

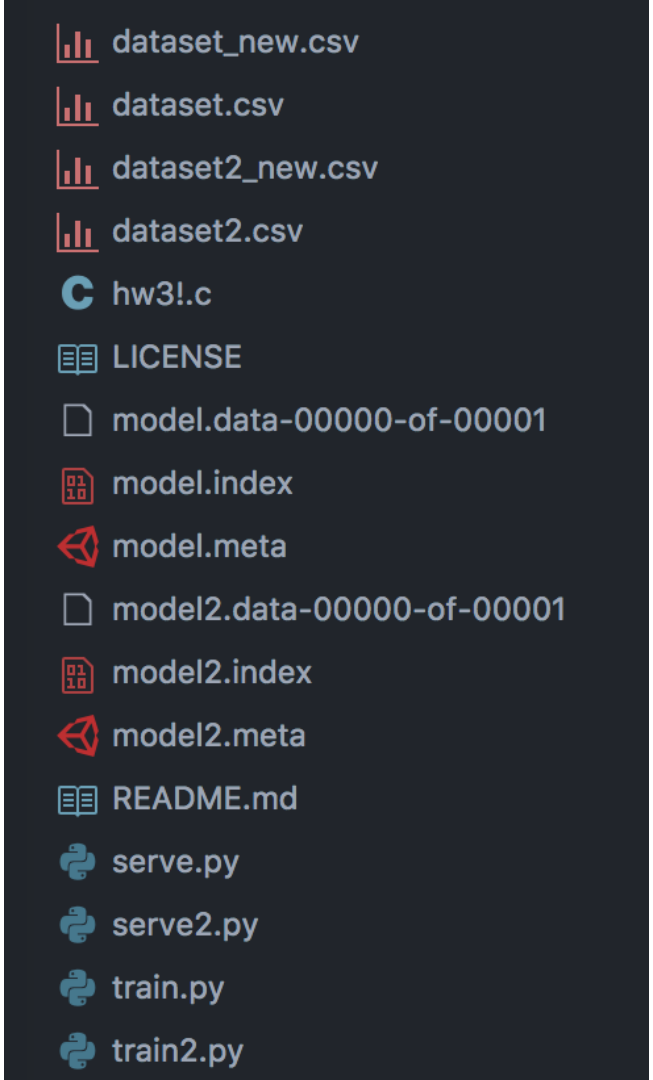
승리한 경우에 대한 상태 값과 베팅 칩 수

dataset.csv

```
18,11,990,990,0,0,0,774
6,16,206,453,0,1321,3,206
8,3,388,312,0,1280,9,388
12,14,1678,302,0,0,10,1323
12,6,12,1268,0,700,7,12
13,1,990,990,0,0,0,897
13,1,93,4,897,986,2,93
15,14,1992,0,0,0,3,1727
7,10,831,540,159,450,2,830
8,7,662,978,0,340,3,531
10,17,111,454,0,1415,7,111
8,1,1775,118,27,60,17,521
14,11,990,990,0,0,0,812
6,20,168,890,0,922,3,168
18,11,1100,880,0,0,4,1039
5,1,990,990,0,0,0,964
16,12,6,460,0,1518,4,6
2,6,1980,0,0,0,16,1197
6,19,990,990,0,0,0,647
6,19,343,97,647,893,2,343
8,5,204,313,0,1463,8,204
2,6,990,990,0,0,0,919
```

Step 2.

서로 다른 두 개의 데이터 셋으로 학습시킨 두가지 모델 생성



Step 3.

두 모델끼리 다시 게임을 진행하여 새로운 데이터 셋을 만듬

dataset_new.csv

```
6,14,654,654,336,336,2,336
20,19,517,849,255,359,4,359
20,17,540,820,268,352,4,352
17,6,650,650,340,340,2,340
1,7,654,654,336,336,2,336
9,15,1682,298,0,0,0,483
19,9,650,650,340,340,2,340
12,5,1700,280,0,0,0,487
9,1,651,651,339,339,2,339
8,13,653,653,337,337,2,337
11,18,1694,286,0,0,0,486
19,9,650,650,340,340,2,340
7,12,1680,300,0,0,5,474
20,9,650,650,340,340,2,340
2,5,1690,290,0,0,0,475
11,6,651,651,339,339,2,339
20,18,651,651,339,339,2,339
15,20,475,1047,0,458,13,458
4,1,652,652,338,338,2,338
1,6,654,654,336,336,2,336
6,14,654,654,336,336,2,336
2,20,656,656,334,334,2,334
20,16,1678,302,0,0,0,495
12,5,651,651,339,339,2,339
15,3,1698,282,0,0,0,490
12,7,651,651,339,339,2,339
18,5,1678,302,0,0,5,486
7,6,652,652,338,338,2,338
```

dataset2_new.csv

```
17,14,1682,298,0,0,0,492
9,20,621,706,318,335,4,335
1,17,630,704,316,330,11,330
4,9,1674,306,0,0,0,477
19,20,1672,308,0,0,0,494
4,12,1704,276,0,0,0,478
9,16,1684,296,0,0,0,483
3,16,1819,161,0,0,0,479
8,18,1692,288,0,0,0,482
18,17,830,170,359,621,5,359
10,6,1674,306,0,0,0,484
16,10,1670,310,0,0,0,491
12,2,1674,306,0,0,0,486
15,12,1678,302,0,0,0,490
4,15,1682,298,0,0,0,478
1,14,831,175,355,619,5,355
9,10,1674,306,0,0,0,483
7,2,528,840,257,355,9,355
4,5,1676,301,0,0,0,477
2,13,654,318,336,672,3,336
8,1,1682,298,0,0,0,482
12,11,830,171,358,621,5,358
14,11,652,314,338,676,3,338
14,5,1690,290,0,0,0,489
5,2,1676,304,0,0,0,478
2,7,1584,396,0,0,0,473
10,6,653,316,337,674,3,337
6,19,1706,274,0,0,0,480
```


?
Q
&
&
A
!