Deep Q Learning with OpenAI

Feng Jiang, Yizhe Hu, Brandon Klein, Zijie(Jay) Wang

Background

Q Learning vs. Traditional NN with CartPole

Policy Based Gradient with CartPole

Deep Q Learning with Breakout

Future Work

OpenAI Gym

“Environments”

observation

Environment-specific object representing the observation of the environment.

Most likely the raw pixel data from the game.

reward

Amount of reward achieved by the previous action.

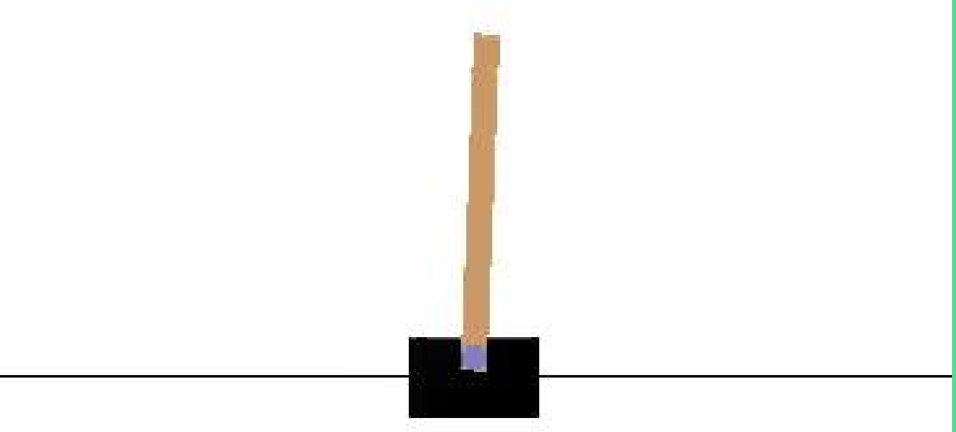
done

Indicates whether a given episode has terminated.

action_space

The actions that can be performed at a given time.

CartPole



observation

Cart position, cart velocity, pole angle, and pole velocity at tip.

reward

Number of frames from the start episode to game termination

done

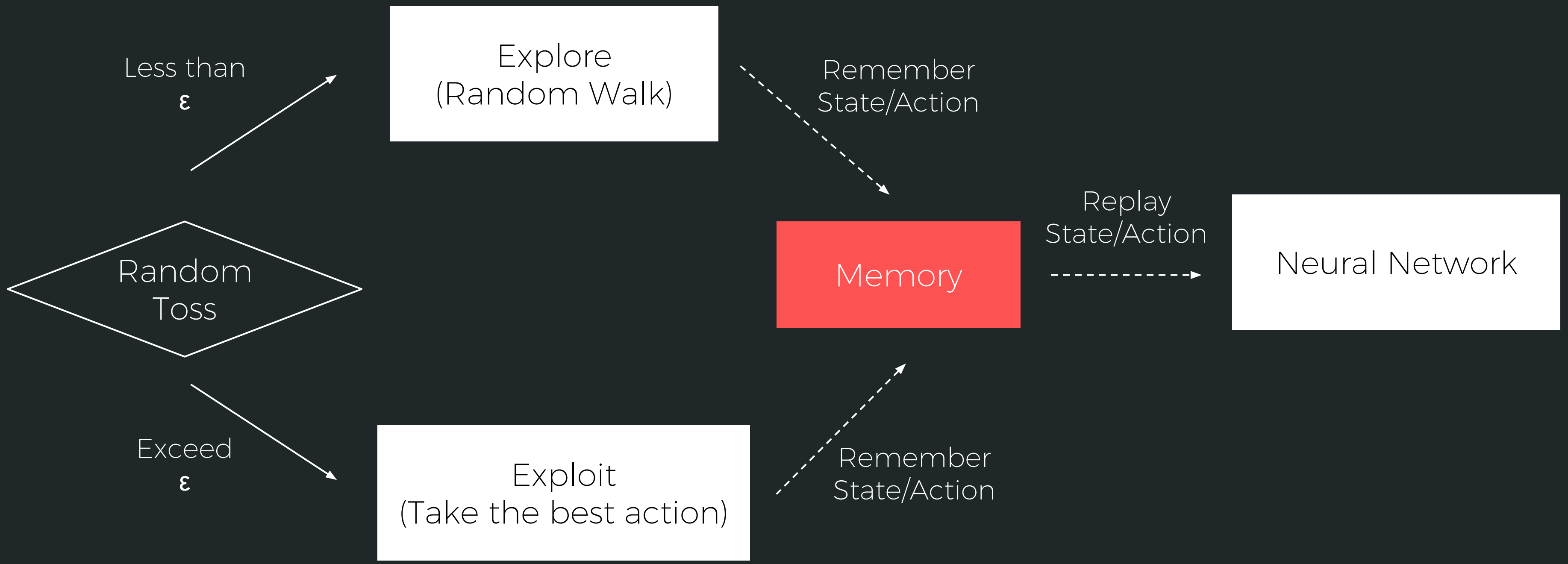
The game is terminated if the pole falls off.

action_space

[-1, 1] indicating moving left or right.

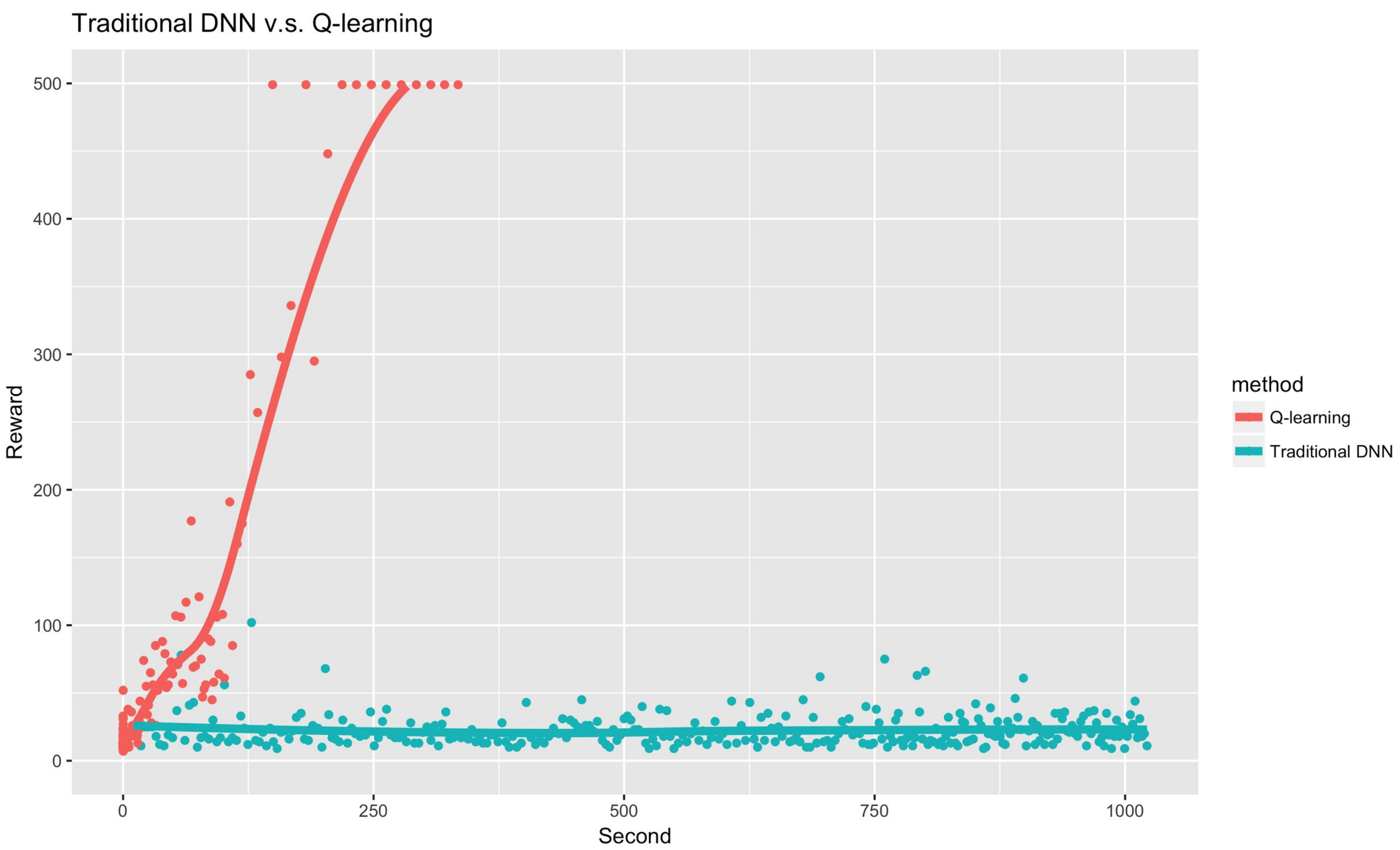
Traditional Neural Network

Q Learning



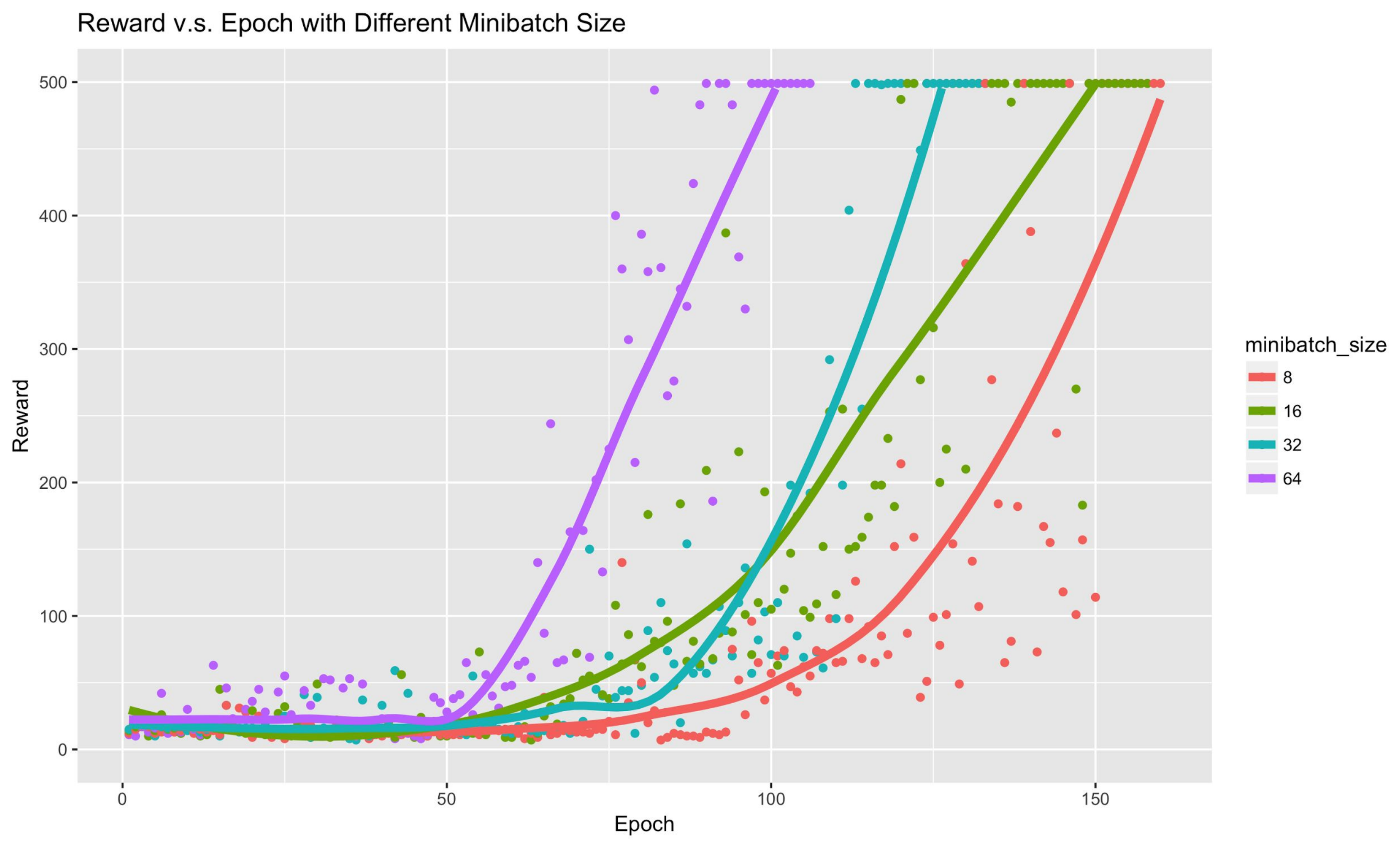
Less than
ε
Explore
(Random Walk)
Remember
State/Action
Random
Toss
Memory
Replay
State/Action
Neural Network
Exceed
ε
Exploit
(Take the best action)
Remember
State/Action

DNN	Q-Learning
5 Layers	3 Layers
128, 256, 512 Units / Layer	24 Units / Layer
Dropout Ratio: 0.2	Dropout Ratio: 0
Activation: RELU	Activation: RELU



Traditional DNN v.s. Q-learning
Reward
500
400
300
200
100
0
0
250
500
750
1000
Second
method
Q-learning
Traditional DNN

Experiments

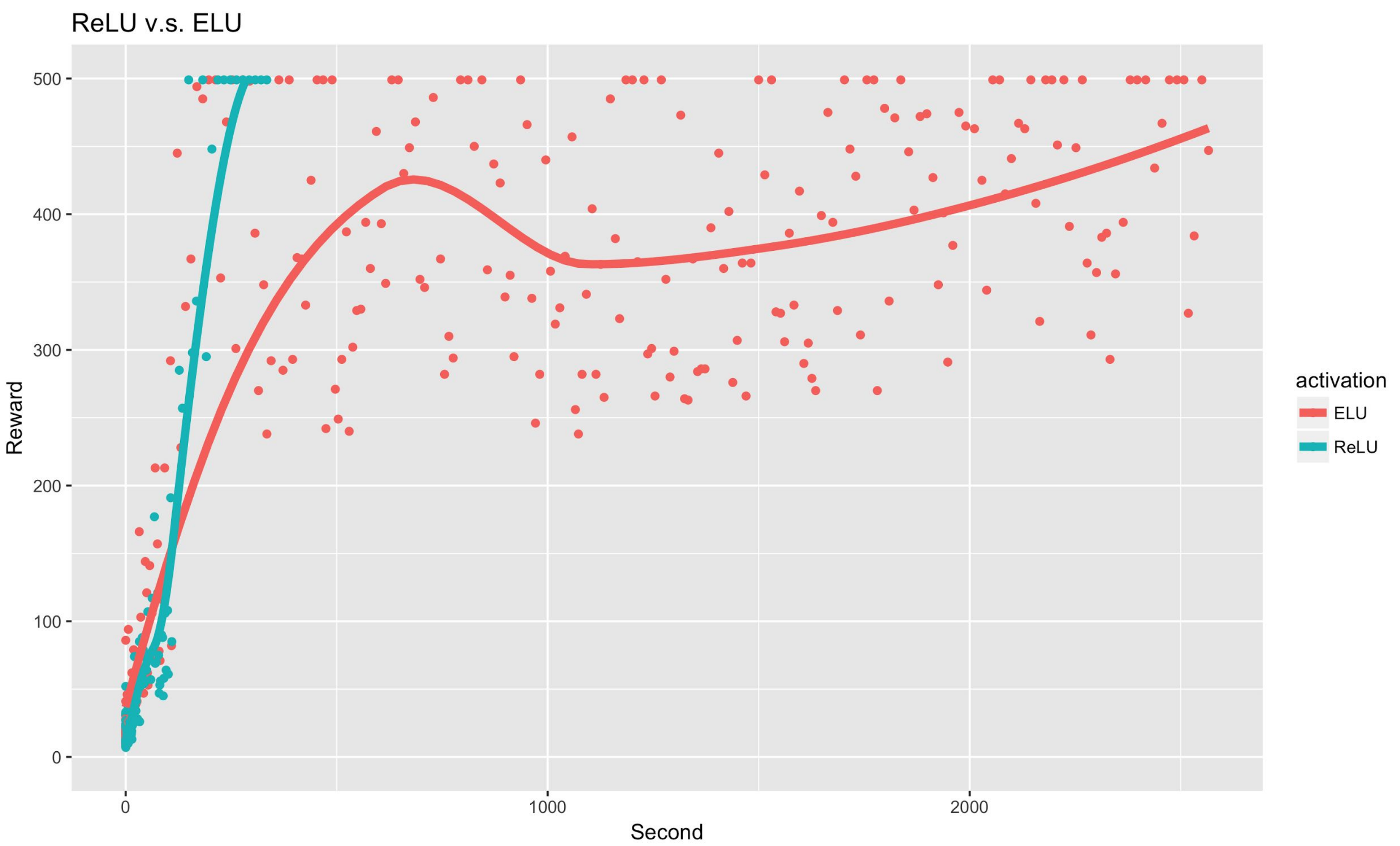


Reward v.s. Epoch with Different Minibatch Size
Reward
500
400
300
200
100
0
0
50
100
150
Epoch
minibatch_size
8
16
32
64

Reward v.s. Epoch with Different Memory Szie
Reward
Epoch
0
100
200
300
400
500
0
50
100
150
200
memory_size
1000
2000
3000
4000
10000

ELU Activation Function

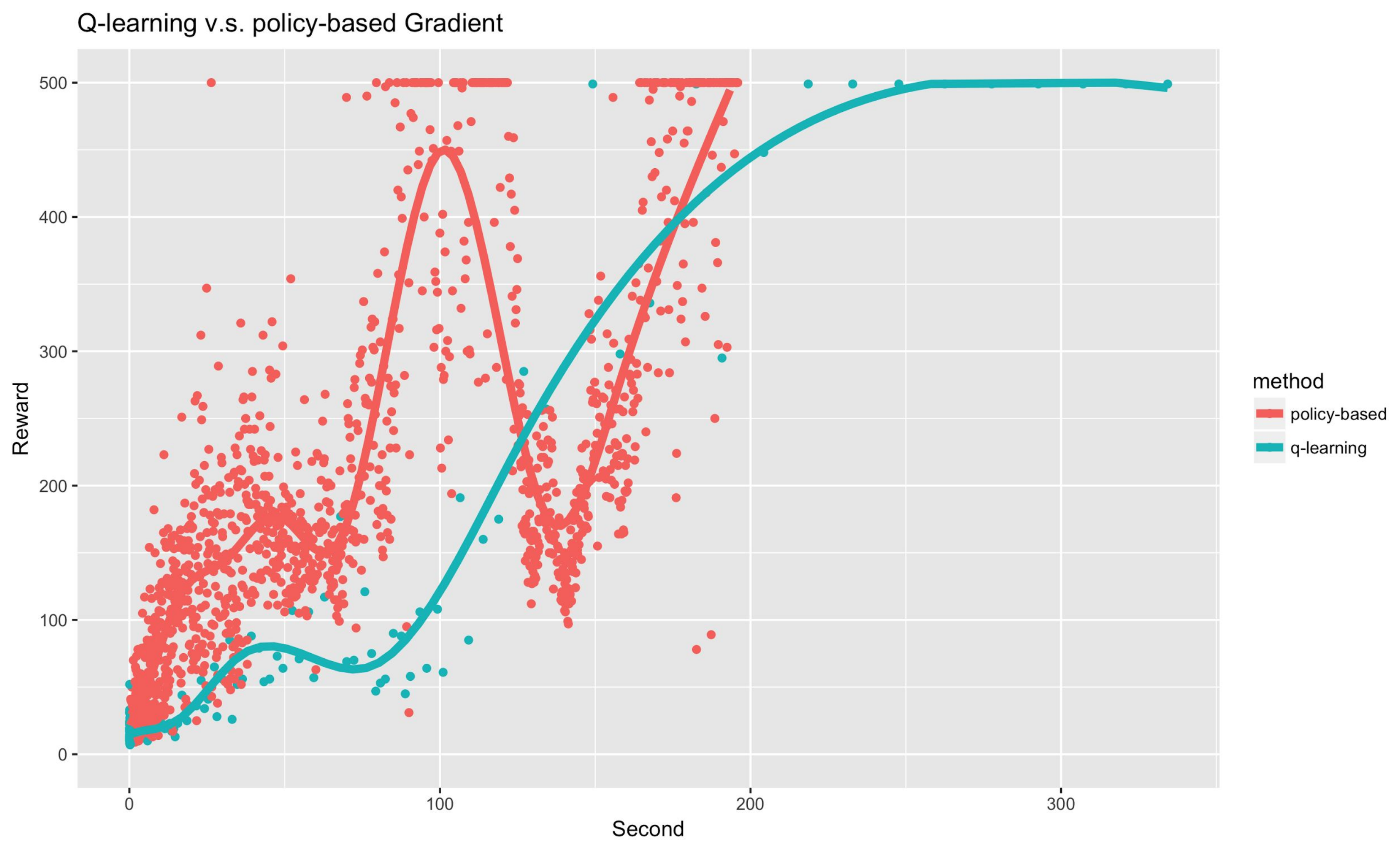
$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha\left(\exp(x) - 1\right) & \text{if } x < 0 \end{cases}$$


ReLU v.s. ELU
500
400
300
200
100
0
0
1000
2000
Second
Reward
activation
ELU
ReLU

Policy-Based Gradient

$$\theta_{t+1} = \theta_t + \alpha R_{t+1} \nabla_\theta \log \pi_{\theta_t}(A_t | S_t)$$

- Monte-Carlo Policy Gradient
- Some environments have easy policy and complex score
- High variance

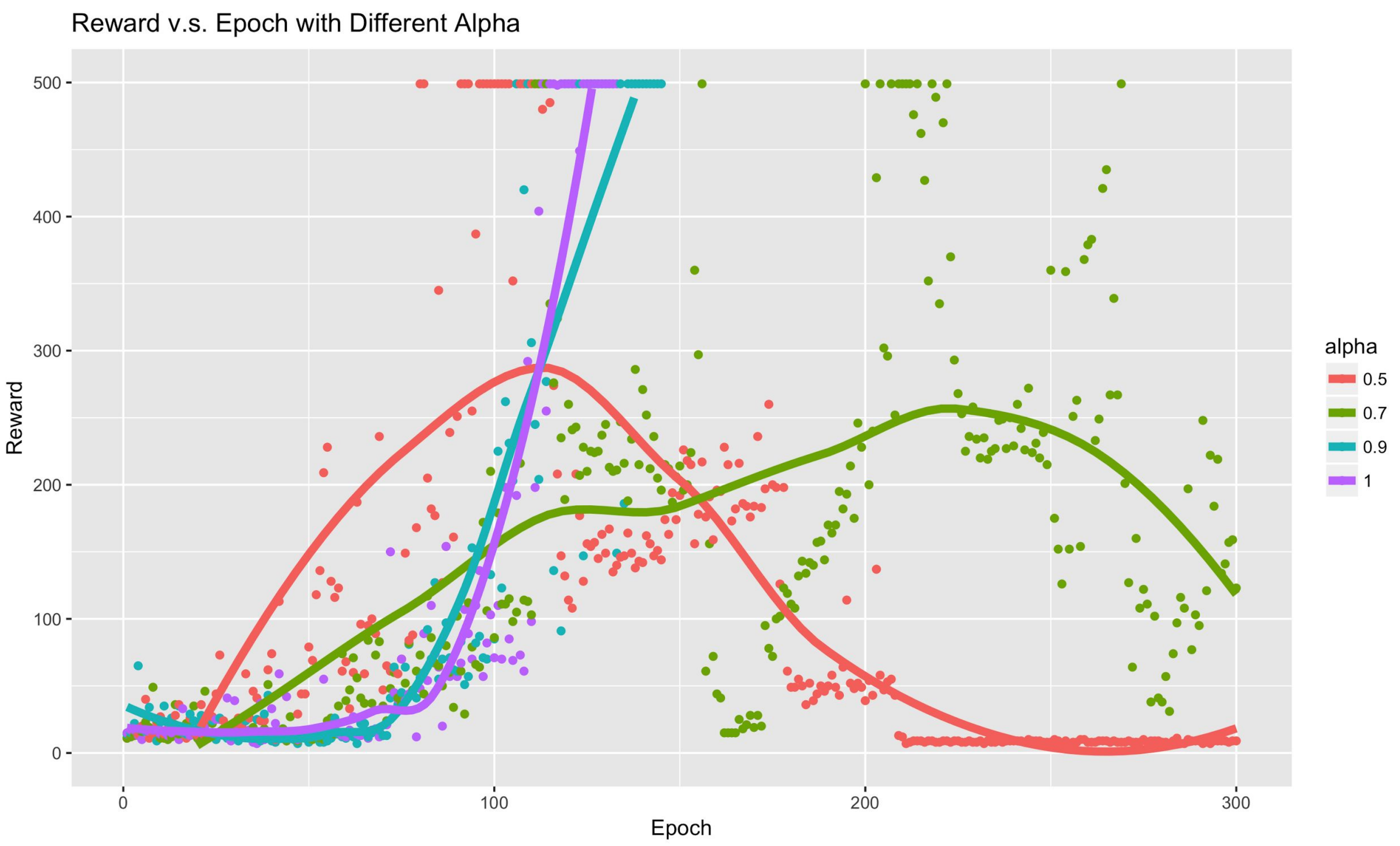


Q-learning v.s. policy-based Gradient
Reward
Second
0
100
200
300
400
500
0
100
200
300
method
policy-based
q-learning

Reward v.s. Epoch with Different Learning Rate
Reward
Epoch
0
100
200
300
400
500
eta
5e-05
0.001
0.005

Use the Old Q Value...

$$y = (1 - \alpha)\hat{Q}(s_0, a) + \alpha \left(r + \gamma \max_{a'} \hat{Q}(s_1, a') \right)$$



Reward v.s. Epoch with Different Alpha
Reward
Epoch
alpha
0.5
0.7
0.9
1
0
100
200
300
400
500
0
100
200
300

Breakout

observation

Raw pixel value of the game frame.

reward

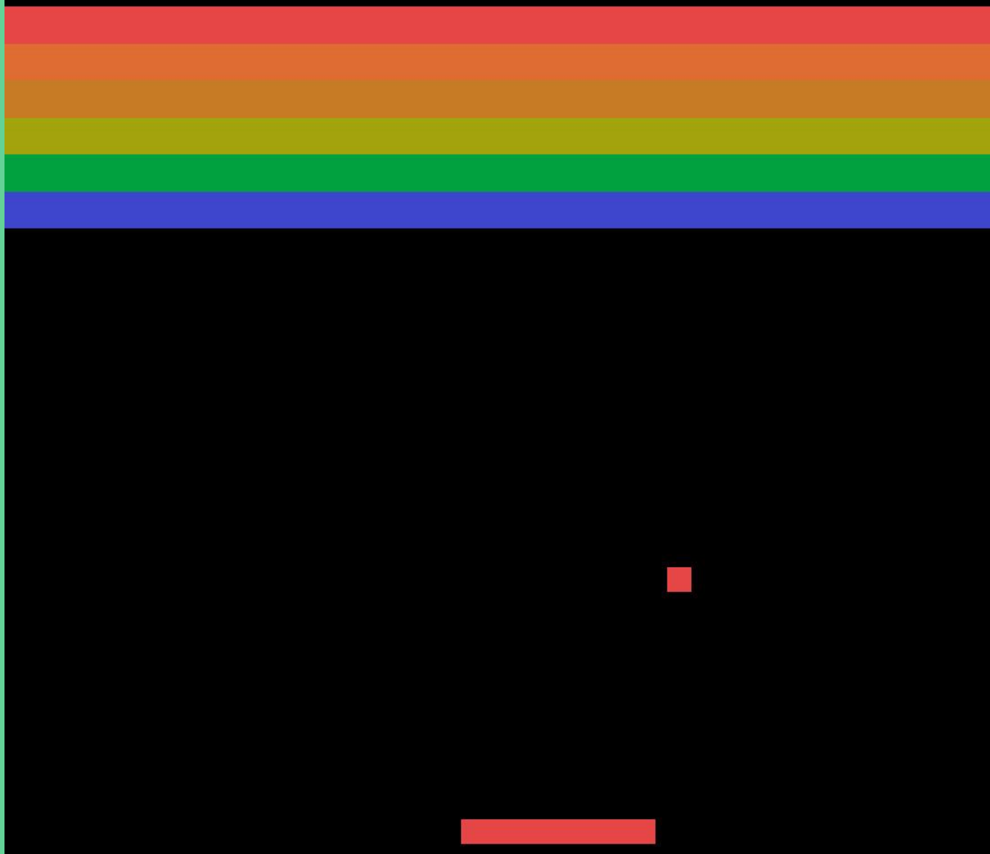
Scores gained by hitting the bricks.

done

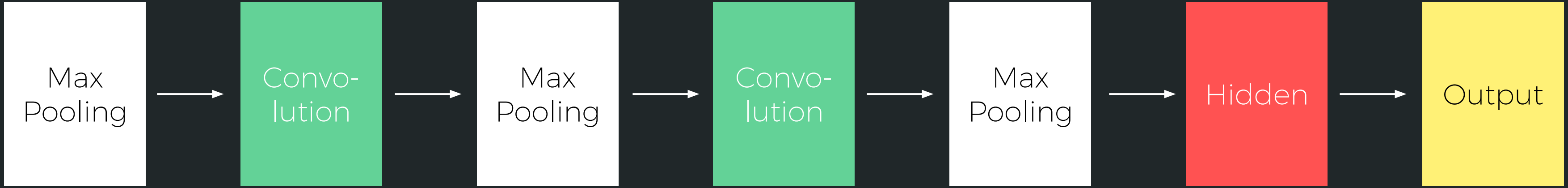
The game is terminated if the ball falls off or all bricks are broken.

action_space

[fire, fire left, fire right, left, right, noop]



Q Learning with CNN



Running on Google Cloud (with GPU) right now...

Future Work

Generalize and Apply to other games

LSTM

Asynchronous Method