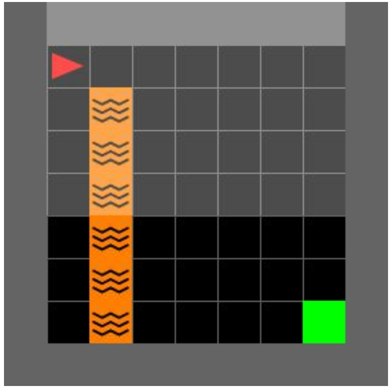
Replay Experience Prioritized DQN Model

—A Performance Measurement in Atari Breakout Game

By Yanzheng Wu, Bofu Dong, Luxi Liu

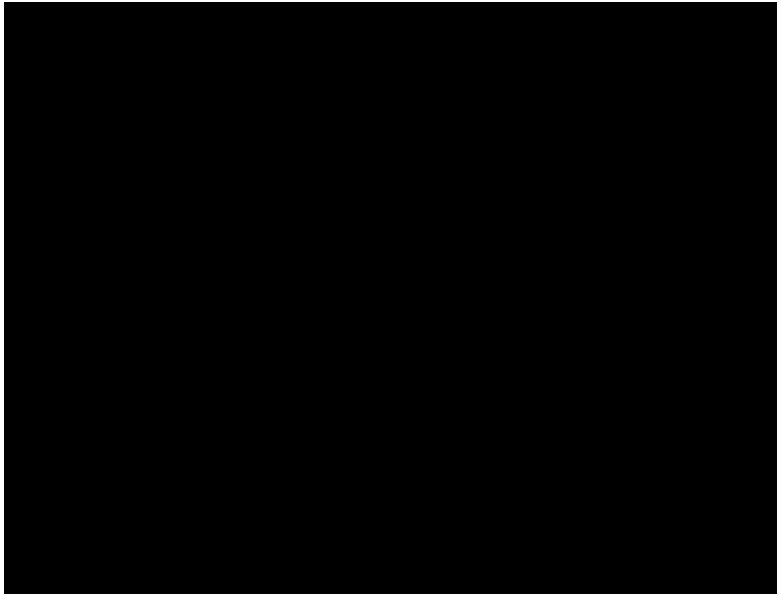
Tufts University

Initial Proposal(Resource Maximization)



Minigrid 7x7

- **Gym MiniGrid**, We tried but we failed :(
    - Failed implementing it due to deprecated packages and dispendency issues.
    - Initial Basic Q-learning method performance:



Q-learning (off policy TD control)

Num	Name	Action
0	left	Turn left
1	right	Turn right
2	forward	Move forward
3	pickup	Unused
4	drop	Unused
5	toggle	Open door
6	done	Unused

Introduction

- We are implementing DQN with **prioritized experience replay** to solve **Atari Breakout** game.
- Games are **not only** entertainment. Training a virtual agent to outperform human players can teach us how to optimize different processes in a variety of different and exciting subfields.
- Tasks **in real life**, such as driving a car, requires the human brain to take in a lot of visual information. A new study from Caltech compares brain scans of humans playing classic Atari video games to AI networks trained to play the same games, discovering that activity in the artificial "neurons" in the AI looked quite similar to activity in the human brain.
- The reinforcement-learning framework alone does not adequately describe decision-making in larger and more complicated tasks. The DQN combines the classic reinforcement learning framework with convolutional neural network, making the system more powerful to detect visual features.

Huge Shoutout to:

- Credit: DQN with Gym Library Inspired by Eugenia Anello(https://www.linkedin.com/in/eugenia-anello/)
- Credit: Replay Buffer Implementation Inspired by Fabio M. Graetz, Ph.D.(https://medium.com/@fabiograetz)
- Credit: Additional interesting RL environments (https://github.com/openai/baselines)

The Atari Breakout Environment

Could this be the one?

Num	Action
0	NOOP
1	FIRE
2	RIGHT
3	LEFT

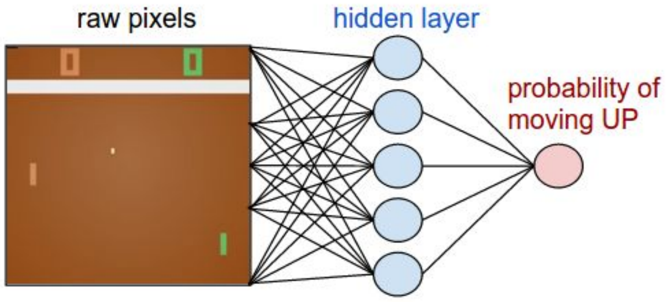
[0,17] values within the Action Space

Action Space	Discrete(18)
Observation Space	(210, 160, 3)
Observation High	255
Observation Low	0
Import	`gym.make("ALE/Breakout-v5")`

Methods

- Deep Q-learning Network using Regular Prioritized Experience Replay
- We actually implemented a fundamental Q-learning algorithm on Minigrid, *and we strongly agree the Atari Breakout has even more State-action Pairs*
- Our inputs are 210x160x3 single frames, we applied reshape and grey-scale for our neural network(84x84x1).

Example of Simple DQN



Double NNs DQN with Prioritized Experience Replay

Initialize network Q
Initialize target network $\hat{Q}$
Initialize experience replay memory D
Initialize the *Agent* to interact with the Environment
while *not converged* **do**

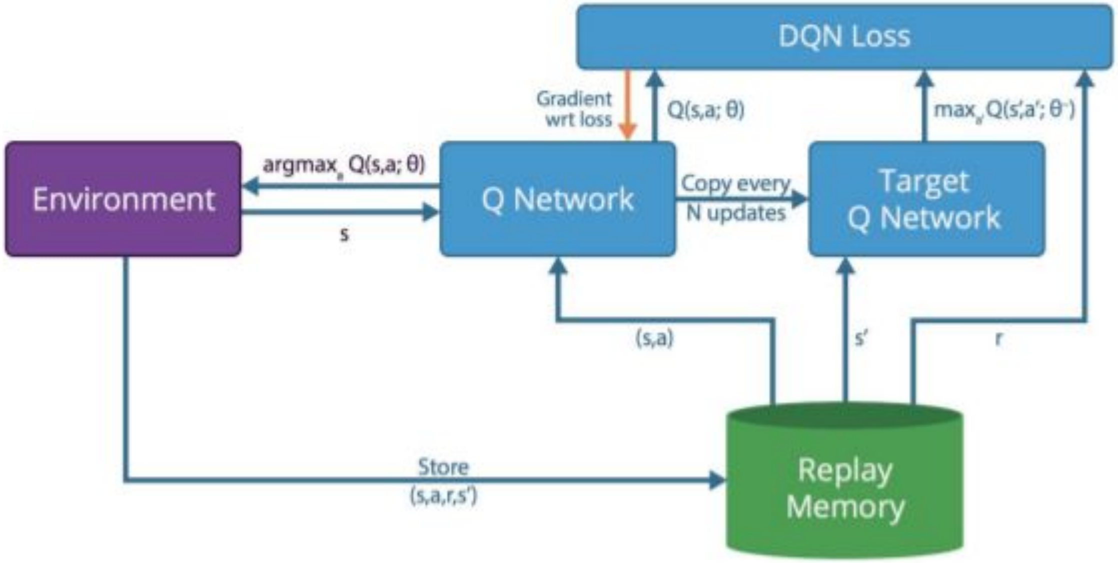
    /* Sample phase
    $\epsilon \leftarrow$ setting new epsilon with ϵ-decay
    Choose an action a from state s using policy ϵ-greedy(Q)
    Agent takes action a, observe reward r, and next state s'
    Store transition $(s, a, r, s', done)$ in the experience replay memory D

    if *enough experiences in* D **then**
        /* Learn phase
        Sample a random *minibatch* of N transitions from D
        for *every transition* $(s_i, a_i, r_i, s'_i, done_i)$ *in minibatch* **do**
            if $done_i$ **then**
                $y_i = r_i$
            else
                $y_i = r_i + \gamma \max_{a' \in \mathcal{A}} \hat{Q}(s'_i, a')$
            end
        end
        Calculate the loss $\mathcal{L} = \cancel{1/N \sum_{i=0}^{N-1} (Q(s_i, a_i) - y_i)^2}$
        Update Q using the SGD algorithm by minimizing the loss $\mathcal{L}$
        Every C steps, copy weights from Q to $\hat{Q}$
    end
end

$$Loss = \frac{1}{2} \cdot [\underbrace{r + max_{a_{t+1}}(Q(s_{t+a}, a_{t+1}; \theta_{t-1}))}_{\text{target}} - \underbrace{Q(s, a; \theta)}_{\text{prediction}}]^2$$

Source: https://towardsdatascience.com/deep-q-network-dqn-ii-b6bf911b6b2c

Double-NN DQN Model



Dynamic Epsilon:

Eps

ilon initial=1,

Epsilon final=0.1,

Epsilon evaluation=0.0

Source: http://rail.eecs.berkeley.edu/deeprlcourse-fa18/static/slides/lec-21.pdf

NN Specs:

- Four 2-Dimension Convoluted Layers(Relu for all activation functions)
  - 32x32, 64x64, 64x64, 1024x1024;
  - Stride 4, 2, 1, 1
- One Flatten Layer
- Two Dense Layers(Including the output layer which contains the probabilities for each Q(s_t, a_i))
- PS. We simply copy the weights of our policy NN to target NN

Problem formulation

Action Space	Discrete(18)
Observation Space	(210, 160, 3)
Observation High	255
Observation Low	0
Import	`gym.make("ALE/Breakout-v5")`

Action Space:

Num	Action
0	NOOP
1	FIRE
2	RIGHT
3	LEFT

Altered Reward function from Bellman :

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha.[r + \gamma.max_{a_{t+1}}(Q(s_{t+1}, a_{t+1})) - Q(s_t, at)]$$

RL training-HyperParams Specs

- Regular experience replay
- Total number of frames to train: 2e7
- Number of frames to validate/evaluate: 1e5
- Mini-batch size = 32(number of frames stacked together for the agent to learn)
- Discount factor = 0.99
- Learning Rate = 0.001
  - (smaller the better(?) but it’s gonna take exponentially more time)
- Reward Params: If Scored:positive reward+1; negative reward -1, otherwise 0.

E.g.: sum(3+1+8-1-1)/5

Train Snippet

- Training

~= 3mins

```
Game number: 000010  Frame number: 00002031  Average reward: 1.6  Time taken: 9.0s
Game number: 000020  Frame number: 00003821  Average reward: 1.0  Time taken: 5.0s
Game number: 000030  Frame number: 00005562  Average reward: 1.1  Time taken: 22.4s
Game number: 000040  Frame number: 00007295  Average reward: 1.0  Time taken: 20.6s
Game number: 000050  Frame number: 00009338  Average reward: 1.7  Time taken: 21.9s
Game number: 000060  Frame number: 00011248  Average reward: 1.4  Time taken: 22.5s
Game number: 000070  Frame number: 00013196  Average reward: 1.4  Time taken: 28.2s
Game number: 000080  Frame number: 00014875  Average reward: 0.7  Time taken: 20.0s
Game number: 000090  Frame number: 00016519  Average reward: 0.7  Time taken: 26.7s
Game number: 000100  Frame number: 00018361  Average reward: 1.2  Time taken: 27.6s
Game number: 000110  Frame number: 00020411  Average reward: 1.7  Time taken: 25.9s
```

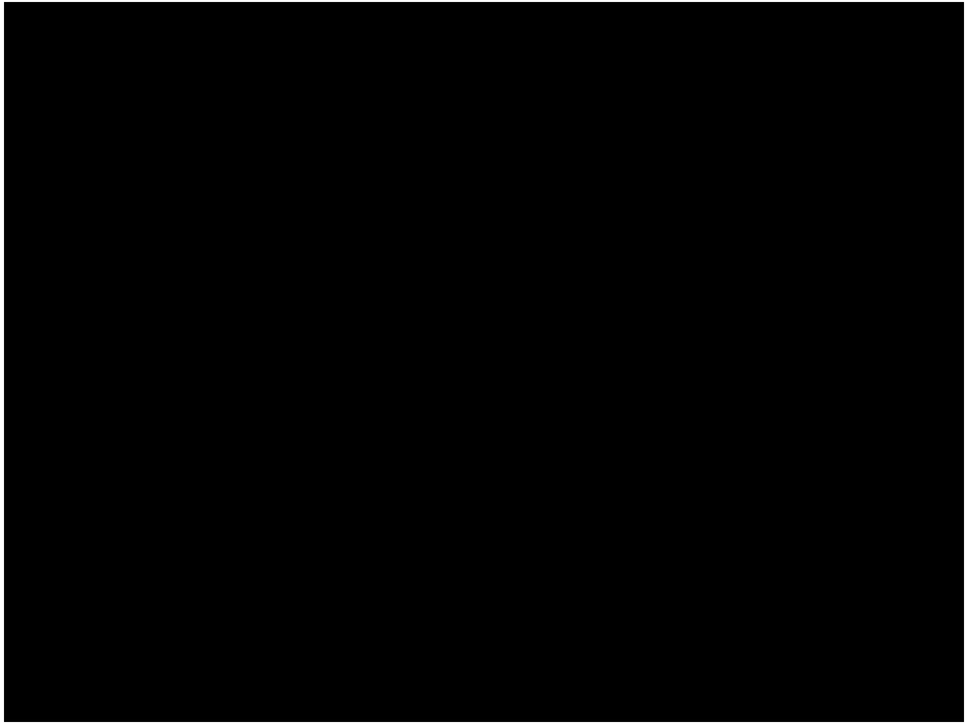
Result: evaluation Snippet

Every 1e5 frames, we evaluate the time it takes to complete 10 games, a GD Loss, Ave. Reward.

```
Game number: 000280  Frame number: 00050096  Average reward: 1.5  Time taken: 10.9s
1/1 [==============================] - 0s 24ms/step

Game number: 000300  Frame number: 00053736  Average reward: 1.0  Time taken: 14.3s
1/1 [==============================] - 0s 19ms/step

Game number: 000400  Frame number: 00071082  Average reward: 0.8  Time taken: 27.9s
1/1 [==============================] - 0s 84ms/step

Game number: 000530  Frame number: 00094905  Average reward: 1.6  Time taken: 84.8s
1/1 [==============================] - 0s 176ms/step
```

~= 3hrs

Result: emulator Snip



Conclusion and take-home message

- Double NN DQN is more efficient use of previous experience, by learning with it multiple times.
- But it comes with great cost: very hard to hypertuning the DQN(to be able to see affective results).
  - Try bigger batch size.
- Time costly to train.
  - Usually takes hours and even days to see results
- Difficult to find the appropriate time to stop training
  - The agent may experience *catastrophic forgetting* when it is trained for too many epochs
  - DQN remembers only success in the buffer, and forget what failure is, then predict everything with high value

References:

- https://github.com/fg91/Deep-Q-Learning/blob/master/DQN.ipynb
- https://leonardoaraujosantos.gitbook.io/artificial-inteligence/artificial_intelligence/reinforcement_learning/deep_q_learning
- https://datascience.stackexchange.com/questions/20535/what-is-experience-replay-and-what-are-its-benefits
- https://www.caltech.edu/about/news/neural-networks-playing-video-games-teach-us-about-our-own-brains
- http://rail.eecs.berkeley.edu/deeprlcourse-fa18/static/slides/lec-21.pdf
- https://towardsdatascience.com/deep-q-network-with-pytorch-and-gym-to-solve-acrobot-game-d677836bda9b
- Tom Schaul, John Quan, Ioannis Antonoglou, David Silver *Prioritized Experience Replay* ICLR 2016.

The End, Thanks!

Tufts UNIVERSITY