Deep Reinforcement Learning Algorithm for Banana Collection game

Shivoh Chirayil Nandakumar[1]
[1]

384 Words
Keywords: DQN, Experience Replay

Student ID:nil
14th January, 2021

1 Introduction

This report provides you an insight on how i implemented the deep learning algorithm and trained the agent for Banana collection game. The agent's action space consists of 4 dimensions and the state space of the environment has 37 dimensions. All the code for this project was developed in Python 3 in the Jupyter notebook.

2 Methods

I used the Deep Q-Network or DQN to train the agent. Moreover, experience replay was used to enhance the efficiency of training. The Optimal Action values (Q^*) are obtained utilising a deep neural network.

$$Q^*(s,a) = \max_{\pi} \mathbb{E}\left[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots \,|s_t = s,\, a_t = a,\, \pi\right],$$

Fig. 1. Optimal Action-Value function

The loss function for training the neural network is as shown below.

$$L_i(\theta_i) = \mathbb{E}_{(s,a,r,s')\sim \mathrm{U}(D)}\left[\left(r + \gamma \max_{a'} Q(s',a';\theta_i^-) - Q(s,a;\theta_i)\right)^2\right]$$

Fig. 2. Loss function

Here the current action values are compared with a target action values. The target action values are updated few frequently.The θ_i represent the current weight in the neural network and the θ_i^- represent the weights used in the target network. We should minimise the value between target action value and the current action value.

As given in the DQN paper published in Nature [1], the psudocode of the DQN is attached below. Please note that we have used 4 state inputs simultaneously for creating a temporal understanding for the model. Thus the preprocessed sequence ϕ_1 represent the four consecutive state inputs combined.

Algorithm 1: deep Q-learning with experience replay.

Initialize replay memory D to capacity N

Initialize action-value function Q with random weights θ

Initialize target action-value function $\hat{Q}$ with weights $\theta^{-} = \theta$

For episode = 1, M **do**

  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$

  For t = 1,T **do**

    With probability ε select a random action a_t

    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$

    Execute action a_t in emulator and observe reward r_t and image x_{t+1}

    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$

    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D

    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D

    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^{-}) & \text{otherwise} \end{cases}$

    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ

    Every C steps reset $\hat{Q} = Q$

  End For

End For

Fig. 3. DQN Pseudo Code

2.1 Deep Neural network Architecture

For this project, i used neural network with four hidden layers containing 64 units, 32 units, 16 units and 8 units respectively. Each hidden layer have an ReLU activation function. The output of the model is the action space for the agent. That is, the output is a four unit layer and the input for the model is a 37 unit state space. Adam optimiser was used for optimising the weights of the Neural Network.

3 Results

The agent was successfully trained to obtain a score of 13+ for 100+ consecutive episodes in less than 800 episodes.

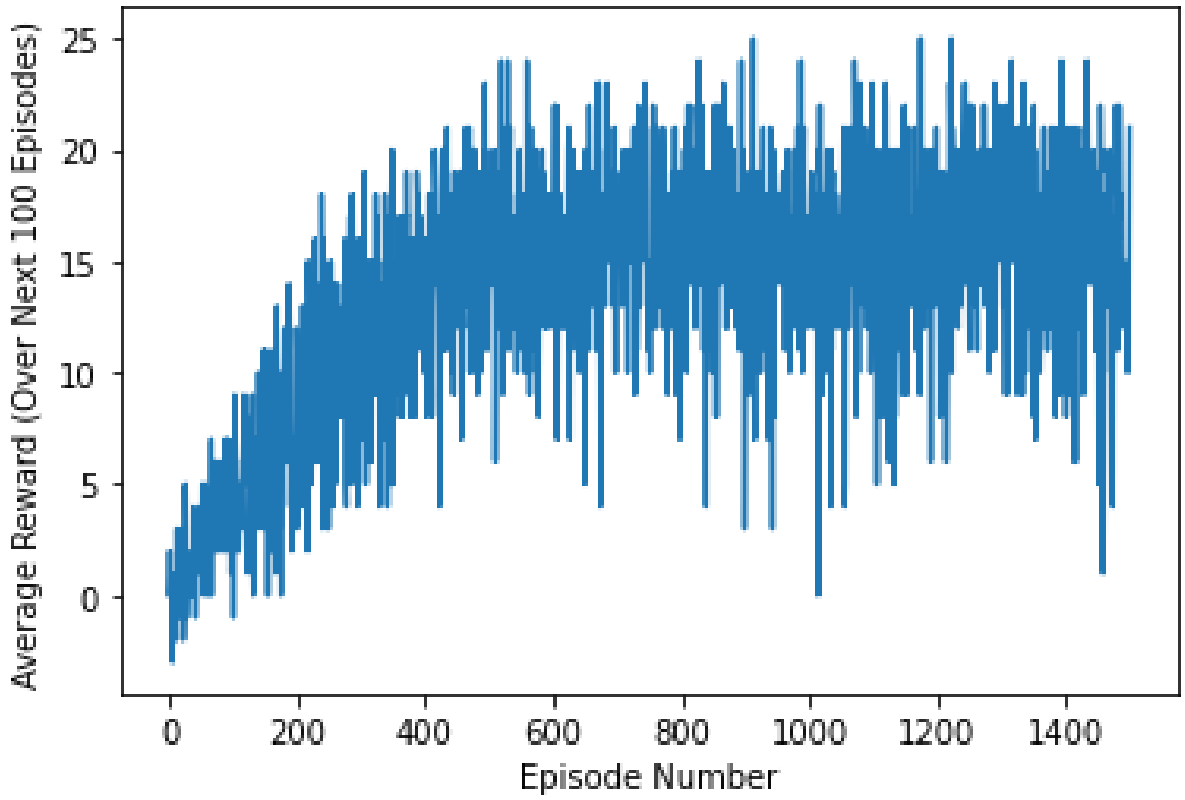

Fig. 4. The successful training session of DQN Agent

4 Future Work

Fine tuning various hyper parameters of the system can possibly improve the efficiency of the system. Adding more hidden layers and changing the number of the units of the layers found to have impact on the training speed of the agent. Moreover, if there is a possibility of high bias, its recommended to use the dropout feature. The Double DQN as well as the combination of various algorithms known as rainbow DQN algorithm can be tried for this project which have a high potential to improve the effectiveness of the agent.

5 References

1. Mnih, V. *Human-level control through deep reinforcement learning*. 2015.