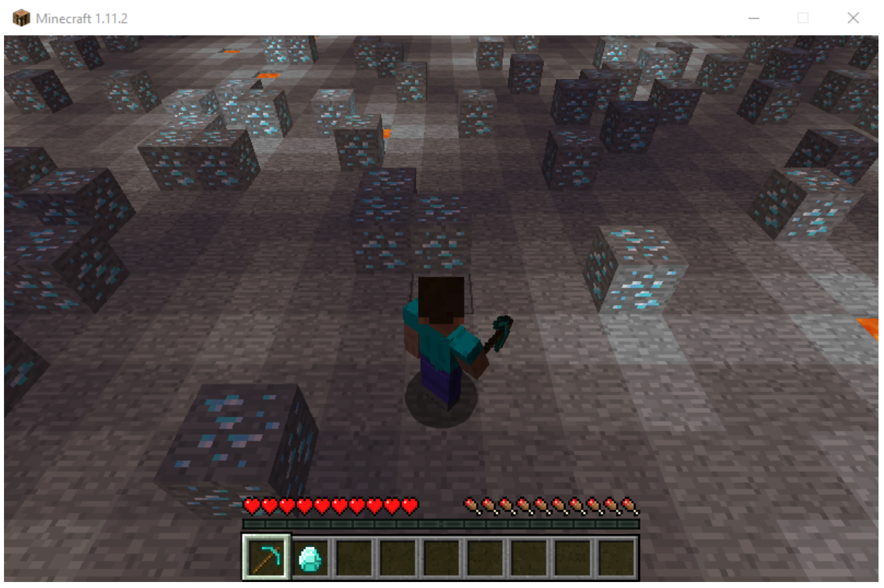
CS 175: Project in AI (in Minecraft): Fall 2020

Assignment 2: Diamond Collector

Sameer Singh (with Kolby Nottingham)
https://canvas.eee.uci.edu/courses/30925

In this assignment we provide a Minecraft agent that uses the DQN reinforcement learning algorithm to learn how to find and collect diamonds. Right now, he spawns in an empty field. You will be tasked with spawning diamonds for the agent to collect, adding obstacles for the agent, and experimenting with the DQN algorithm.

Note: Running `assignment2.py` to completion takes about 40 minutes, so start this assignment several days early. If you have access to GPU acceleration, feel free to modify the code to use CUDA, but you will not get amazing speedups. The Q-Network is so small that the real bottleneck is Malmo.



1 Task Description

Steve has a diamond pickaxe and wants to continue to craft his inventory of diamond armor and tools. The thing is, Steve can only see two squares in any direction. He needs to learn how to navigate to diamond ore while avoiding any hazards he may face along the way.

We will be modeling this scenario using Malmo. We provide you with the reinforcement learning agent, Steve, in `assignment2.py`. The Deep Q-Network (DQN) learning algorithm is already implemented in the file. The purpose of this assignment is to familiarize yourself with a reinforcement learning algorithm and the Malmo documentation. Your changes to the Malmo mission should end up looking similar to the screenshot above.

1.1 Overview of the Code

The python file `assignment2.py` contains a number of functions for building a Malmo mission and using the DQN learning algorithm. It also contains a class for the Q-Network itself. Hyperparameters for Malmo and the DQN algorithm can be found at the top of the file. The code also regulary logs your agent's returns in a text file and a graph in a png file called `returns.txt` and `returns.png` respectively. You will not be required to modify most of these functions. Any functions you do modify should be included in your report with an explanation of your modification.

1.2 Setup and Running the Code

Before running the code for this asssignment, you will need to pip install `numpy`, `matplotlib`, and `torch`. The first two should easily install with `pip install numpy matplotlib`. Depending on your setup, pytorch will require a different installation process. Follow the installation instructions at https://pytorch.org/get-started/locally/. Once that's done, all you need to do to run this assignment is to copy the `assignment2.py` file above to the `Python_Examples` folder, and after launching Minecraft, run `python assignment2.py`. If everything runs successfully, the agent should be moving around. The output in the terminal should include a progress bar and several statistics. Please do this as soon as possible, and post on Campuswire if unable to do so.

1.3 Exploration Strategies

When you run the program now, the agent will probably prefer one or two action rather than moving about randomly. This is because the agent is executing the action with the highest predicted Q-value from the Q-network. Because the Q-network is untrained, we'd much rather the agent move randomly at first to explore the space. Then once Steve has learned more, he can start acting according to the Q-network.

Modify `get_action` to change the agent's policy from greedy to ϵ-greedy. This means the agent will move randomly with probability ϵ and greedily otherwise. ϵ will then be decreased as training continues. We provide epsion as one of the inputs to `get_action`.

1.4 Spawning The Diamonds

```
<DrawCuboid x1='{}' x2='{}' y1='2' y2='2' z1='{}' z2='{}'
type='air'/>".format(-SIZE, SIZE, -SIZE, SIZE)
<DrawCuboid x1='{}' x2='{}' y1='1' y2='1' z1='{}' z2='{}'
type='stone'/>".format(-SIZE, SIZE, -SIZE, SIZE)

YOUR CODE GOES HERE

<DrawBlock x='0' y='2' z='0' type='air' />
<DrawBlock x='0' y='1' z='0' type='stone' />
```

Your first task is to spawn diamonds for Steve to mine. The `GetMissionXML` function returns an XML string that specifies the mission. Inside `Mission->ServerSection->ServerHandlers->DrawingDecorator` there are two `DrawCuboid` lines that reset the map and two `DrawCube` lines that ensure the agent is standing in an available spot on top of a stone block. These lines are shown in the code block above. After the map is reset and before the agent's starting position is prepared, you will add tags that spawn diamond ore in the y=2 plane. These blocks should be placed randomly so that each time `GetMissionXML` is called the blocks will be in different places. We provide a hyperparameter called `REWARD_DENSITY` that represents the suggested probability of placing diamond ore at any given location.

Before Steve will learn to mine diamonds, we need to give him a reward signal for him to optimize. Add a reward tag to the AgentHandlers in the mission xml that will give Steve +1 reward each time he picks up a diamond. When you run the program, the progress bar should show non zero rewards after each episode.

Once these blocks are placed, run the python file. The agent should begin to learn to navigate to the diamond ore and collect it. Watch the `returns.png` file as the agent trains. After about 2,000 steps the return should be obviously increasing. This is how you know the agent is working properly. Run `assignment2.py` to completion and save the generated plot and text files. Be sure to change the name of the files or save them in another directory so that you don't accidentally overwrite them.

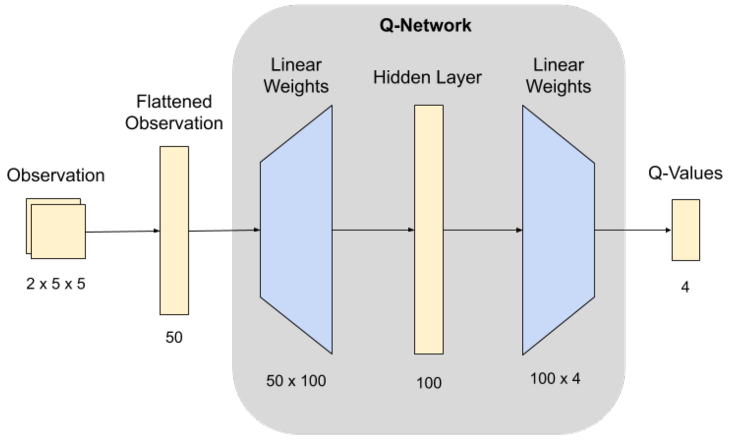
Refer to https://microsoft.github.io/malmo/0.30.0/Schemas/Mission.html for documentation on editting the Malmo mission xml.

1.5 The Floor Is Lava!

Now lets make things more interesting for Steve. Your next task is to add lava in the floor. Randomly place lava in the y=1 plane. We suggest placing the lava block with the probability specified by `PENALTY_DENSITY`. Your Malmo environment should now look similar to the screenshot above.

Finally, we will be penalizing Steve for stepping on your lava blocks by giving him negative reward. Add a reward tag to the AgentHandlers in the mission xml to penalize the agent for falling into lava. A reward of -1 should work. Run `assignment2.py` to completion and save the generated plot and text files.

1.6 Q-Network

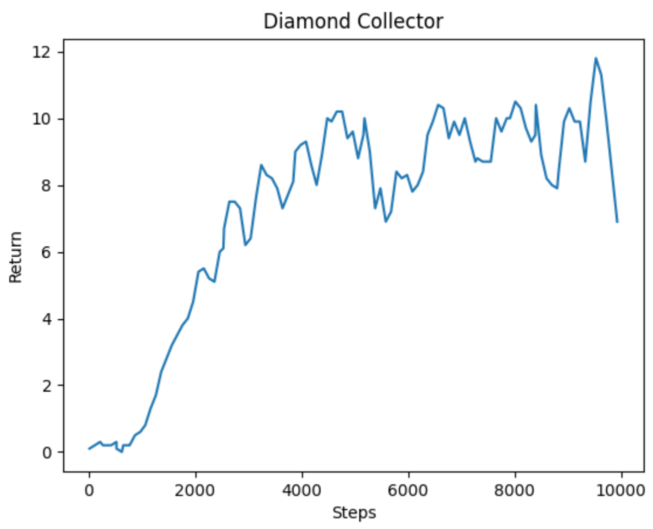


Right now the Q-Network uses two fully connected layers with a ReLU activation function to calculate Q-values. The above diagram illustrates the current Q-Network. Yellow block represent your data, and blue blocks represent the network weights. This is not a very interesting network. Modify it to use additional layers, a different activation function, or convolutional layers. Remember to make sure you are outputting the correct size tensor from the forward function. Run `assignment2.py` to completion and save the generated plot and text files.

2 What Do I Submit?

Here we'll describe what exactly you need to submit to the assignment on Canvas. For the return graphs, the graph should be obviously improving as the number of steps increases, but you do not need to achieve a specific return value. Also if you are running out of time you can submit incomplete return graphs, but you will be docked points. A complete return graph should go up to at least 10,000 steps. An example graph is provided below.

1. **Code: get_action function (20 points):** The `get_action` function should be modified to implement an ϵ-greedy policy.
2. **Code: GetMissionXML function (20 points):** The `GetMissionXML` function should be modified in 2 ways, worth 10 points each. First, diamond ore should be spawned and a positive reward added for collecting it. Second, lava should be spawned with a negative reward for falling into it.
3. **Code: QNetwork class (20 points):** Modify the Q-Network and show us what you changed.
4. **Graph: Diamonds Only Graph (10 points):** Provide the graph/png output of your original experiment mining for diamonds without lava.
5. **Graph: Diamonds and Lava Graph (10 points):** Provide the graph/png output of from mining for diamonds with lava.
6. **Graph: Q-Network Graph (10 points):** Provide the graph/png output of from mining for diamonds with lava.
7. **Extra Credit: Improving the Agent (20 points):** Modify the algorithm or tune hyperparameter so that the agent either trains in less steps or reaches a significantly higher return. If successful, share your new graph, modified code, and explain why the changes you made helped.
8. **Comments:** Any comments about your submission that you want to bring to our attention as we are grading it. This is completely optional, I expect most of you to leave this empty.
9. **Statement of Collaboration (10 points):** It is **mandatory** to include a *Statement of Collaboration* with respect to the guidelines below. Include the names of everyone involved in the discussions (especially in-person ones), and what was discussed. You should also include the links to all online resources you used for the assignment in this section.



All students are required to follow the academic honesty guidelines posted on the course website. For programming assignments, in particular, we encourage the students to organize (perhaps using Campuswire) to discuss the task description, assignment requirements, bugs in our Malmo code, and the relevant technical content *before* they start working on it. However, you should *not* discuss the specific solutions, and, as a guiding principle, you are not allowed to take anything written or drawn away from these discussions (i.e. no screenshots/photographs, written notes, etc.). The same holds for online resources: you are allowed to read the description of algorithms, but your code should be your own. Especially *after* you have started working on the assignment, try to restrict the discussion to Campuswire as much as possible, so that there is no doubt as to the extent of your collaboration.

Acknowledgements

This assignment was conceived and created by Kolby Nottingham.