Tentative Title: Evaluating DQNs in VizDoom

Students:
Nolan Winsman
Tim Fields
Ethan Poe
Sam Laude

GitHub: https://github.com/nolanwinsman/Team-Doom

Goal: Increase performance in one or more VizDoom scenarios using Deep Reinforcement Learning.

Approach: Train agents using the PyTorch DQL agent included with VizDoom: https://github.com/mwydmuch/ViZDoom

Note: Your project will be graded primarily on the quality of your written technical report, but the quality of the report is obviously limited by what your code can actually do. The following guidelines describe the minimum work required to make achieving a certain grade possible. However, even if your code passes the specified threshold, your grade could be lower if the report does not adequately describe the work you did.

C-/C/C+: Do repeated runs of the out-of-the box VizDoom DQN agent on a domain and characterize both its learning and final performance to firmly establish a baseline.

B-/B/B+: Tweak the DQN agent is some small but meaningful way that produces a measurable difference behavior. Either improve the performance, or demonstrate how performance varies as you change some aspect of the agent's configuration (such as different values for certain hyperparameters).

A-/A/A+: Demonstrate a measurable performance increase in at least one of the Doom scenarios, and also evaluate performance in multiple (at least 3) scenarios.

Report Guidelines: You'll need to start by thoroughly understanding the existing PyTorch DQN code. You will need to adequately describe how it works. You will need to discuss the mathematics of DQNs a bit. You will also need to thoroughly describe the Doom scenarios you are training in, including the rewards, discount factor, available actions, etc. The results will need to show learning curves that average across at least 5 runs for each scenario. You should also evaluate the final agent performance (without learning). The final evaluation data can be presented in tables.

Suggestions: Start collecting data for the baselines now. Make sure you know all parameter settings before you run so you can save that information and talk about it in the report. Everyone should be running trials once you have figured out a good way to collect data. As far

as how to improve the agent, consider various hyperparameters and features of the network architecture. You could add layers or remove layers. You could change the learning rate, or the size of the replay buffer. However, although I do encourage experimenting with various parameters, when it comes time to run an experiment, make sure you only focus on just one or two parameters so that it is clear what is causing improvements in performance.

You have comment privileges on this document. You have until March 30th to request changes to the grading criteria on this document.