UDACITY

MLE Capstone Proposal

Training DRL Actor-Critic Agent(s) to Play Pommerman

Kevin Chen

June 22, 2019

Domain Background

Since the day IBM's Deep Blue beat world's best chess GM Garry Kasparov back in 1997, games have been used to benchmark the progression of AI, especially in comparison to human skills in human-based activities. Varying machine learning models have showcased their utilities through subsequent games, from deep neural networks playing Atari 2600 games to beating the best player at Go.

This project is inspired by a more recent feat where the OpenAI Five defeated five top-ranked players in a game of Dota 2. That is a special milestone because it involves 5 NNs having to collaborate in order to beat their 5 human opponents, where previous games are done by AIs in isolation. This opens up a new AI field of Multi-agent Systems (MAS) that can be used for simulations[1] of real-world autonomous cyber-physical systems, including industrial robotics and autonomous vehicles.

For the Capstone, I am aiming to replicate (to a smaller scale) the success of OpenAI Five using the game Pommerman.

Project Statement

Pommerman[2] is a online variation of Nintendo's Bomberman, where the objective is to beat your opponent(s) by planting bombs, trapping them, and blowing them up in a grid environment with block obstacles, as described in Datasets and Inputs. However unlike the original Bomberman, Pommerman will be played using trained AI-agents.

The goal of this project is to develop my own Actor-Critic-based agent that can beat out the boilerplate agents and came with the Pommerman GitHub source code, whether in a 1v1 or 2v2. If 2v2, my agents will need to cooperate using some Multi-Agent Reinforcement Learning techniques. Additional details about the boilerplate agents can be found in the Evaluation Metrics and plan to defeat them in the the Solution Statement.

Datasets and Inputs

Pommerman's environment is a 11x11 board. For each battle round, although the board is randomly initiated, all the agents (up to 4) will start at a corner.

1 Glockner, Cyrill et al. "Simulators: The Key Training Environment for Applied Deep Learning Environment" *Towards Data Science* Medium post (2018)

2 Link to Pommerman competition site

Each agent can only carry and plant one bomb at a time. Once an agent plants a bomb on a square, the agent cannot plant another one until the bomb has been donated. The time from plant to detonation takes 10 “time steps”. The blast radius is 2 squares in all cardinal directions. If an agent (whether same or opposing team) is in the blast radius when denoted, then the agent is removed from the game. If all of the agent(s) from a side is gone, the other side wins and the battle round ends.

Randomly scattered around the board are wooden and rigid walls, which agents cannot step on. Wooden walls can be removed by a bomb while rigid ones cannot. There are also power-ups that agents can collect to enhances their strengths, from wider blast radius to shorter time-to-detonation.

There are at most 7 actions the agent can take for each “time step”: skip move, move up, move down, move left, and move right, plant bomb, and relay massage to friendly agent (in 2v2).

Just like OpenAI gym, after cloning Pommerman’s source code, I will be able to render the environment and see how the agents play out in a graphical window.

Solution Statements

My solution is to implement a variation of actor-critic DRL model to win the 1v1 and actor-critic multi-agent DRL (MADRL[3]) model to win the 2v2. Planned models include (multi-agent) deep deterministic policy gradient (MADDPG), counterfactual multi-agent (COMA), which can handle the multi-agent credit assignment problem, or even *actor-mimic* methods that enable that enables better multitasking and transfer learning.

Fortunately, the environment is only a small discrete 11x11 grid with a 7 maximum actions choices so I do not have to worry about continuous spaces. Also there are only at most 2 agents per team. However, with the various types walls and power-ups, there will be a lengthy set of possible environment spaces. Nevertheless, do not expect my actor-critic model will be overly sophisticated or significantly “deep”.

Benchmark Models

I will use two of Pommerman’s boilerplate agents as benchmark models. The first is a “simple” agent, which is created using dynamic programming. The second is a “tensorforce” agent, a specialized agent using a proximal policy optimization (PPO) model.

Evaluation Metrics

After training my actor-critic model(s), I will first play 1v1 with the “simple” and “tensorforce” agents, 10 matches each. Each match consist of up to round 999 rounds. Whichever agent wins 500 rounds first

3 Nguyen, Thanh Thi et al. “Deep Reinforcement Learning for Multi-Agent Systems: A Review of Challenges, Solutions and Applications” *arXiv preprint arXiv:1812.11794v2* (2019).

wins the current match. If I can get my actor-critic agent to win 7+ matches, especially in a resounding fashion, then I can consider that a success. I will then proceed to do the same for the 2v2.

Project Design

The first step is to clone Pommerman and set up my local environment that enables me to experiment and plug-and-play on agent models and test them on a rendered executable game. Fortunately, Pommerman made that easy to do.

The second step is to do deeper research on which DRL and MADRL models to try out on the agents. I have some ideas mentioned in Solutions Statements but those may be subject to change once I do more research.

The third step is to find the computing environment to train agents. I will try try to AWS and Google Compute GPU cloud environments since I have credits for both.

Once I have the trained the model(s), the fourth and final step is to run them against my Benchmark Models and log how fared my models fared. If my model(s) if in is within the ballpark of my Evaluation Metrics, then I will go back to the second step and further tweak my models.

I repeat this process until I have something that I feel confident to report on my final Capstone Project, even if my initial criteria has not been met.