Games either have complete information (such as chess where everything is visible), or incomplete information such as the real world (not everything possible is known). There are a lot of games being tested in RL (reinforcement learning) with all actions visible such as Cartpole and Blackjack in places such as the Open AI Gym Arcade environments (Brockman et al.). Games such as Pong and Seaquest have been trained with many algorithms, and an early algorithm has been DQN (Deep Q-Network) (Mnih et al.). A lot of research with complete information has been done and fascinating science, such as how neuronlike elements can be used with Cartpole – a game where a pole is balanced (Barto et al.).

Training on games with incomplete information has been less successful than with complete information. From the MIT Technology review, playing poker required information similar to the real world – it was incomplete – and that's what makes it fascinating (Knight 2017). I am making a game that does not seem to have been tested before specifically with DQN, by finding how an agent can use incomplete information to win a two player game of BS. BS has incomplete information, because the agent doesn't know what specific cards the opponent has, and there are different potential results of the agent seeing similar information and choosing an action. The agent has to decide when to call "bs", and to find good situations to do so. There have been card games already made such as poker and even toolkits for reinforcement learning in card games such as Bridge or UNO (Zha et al.). Each card game is unique, and I would like to find out how to train an agent on BS.

I coded a two player BS card game and I am right now working on how to train an agent. I plan to use DQN initially, since it appears to be one of the basic algorithms, and then later change to another algorithm if applicable such as PPO (Proximal Policy Optimization). I'm using the framework in Open AI Gym to create a custom environment for my agent (Brockman et al.).

The game is simplified to have two players, and in each game a certain random amount of cards are taken out of the deck in order for the players to not inherently know what cards the opponent has. With Open AI Gym, the agent has an observation, and I plan to simplify the observation space by giving an array of values, but not specifying the specific card. For example, if the agent has two threes, one four, the agent will see [2,1,etc..] when the agent has to put down threes, and [1,etc..0] when it has to put down a four. I have also created a scripted agent for my agent to play: the agent tells the truth when possible, and currently says "bs" randomly one out of every four times. I hope to gain data from training with DQN, and find a way such that the agent can win the majority of the time against the scripted player.

Work Cited

Barto, Andrew G., Richard S. Sutton, and Charles W. Anderson. "Neuronlike adaptive elements that can solve difficult learning control problems." *IEEE transactions on systems, man, and cybernetics* 5 (1983): 834-846.

Brockman, Greg, et al. “[1606.01540] OpenAI Gym.” *arXiv*, 5 June 2016, https://arxiv.org/abs/1606.01540. Accessed 2 January 2024.

Mnih, Volodymyr, et al. “Playing Atari with Deep Reinforcement Learning.” *arXiv*, 19 December 2013, https://arxiv.org/abs/1312.5602. Accessed 2 January 2024.

Knight, Will. “Why Poker Is a Big Deal for Artificial Intelligence.” *MIT Technology Review*, 23 January 2017, https://www.technologyreview.com/2017/01/23/154433/why-poker-is-a-big-deal-for-artificial-intelligence/. Accessed 4 January 2024.

Zha, Daochen, et al. “[1910.04376] RLCard: A Toolkit for Reinforcement Learning in Card Games.” *arXiv*, 10 October 2019, https://arxiv.org/abs/1910.04376. Accessed 10 January 2024.