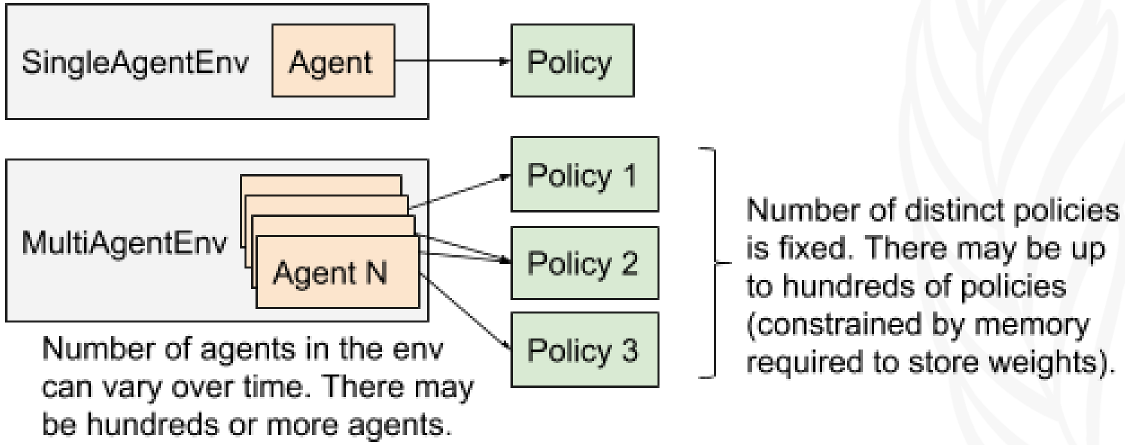
CSE546 PROJECT: MARL WITH NFSP

Nikhil Vasudeva

University at Buffalo The State University of New York

Multi-Agent Reinforcement Learning

- Multi-agent environments where more than 1 agent are present and can interact with each other or can influence the environment.
- In the setup being proposed, each agent has its own policy.

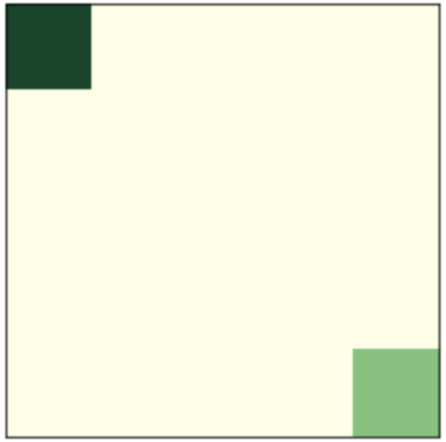


Number of agents in the env can vary over time. There may be hundreds or more agents.

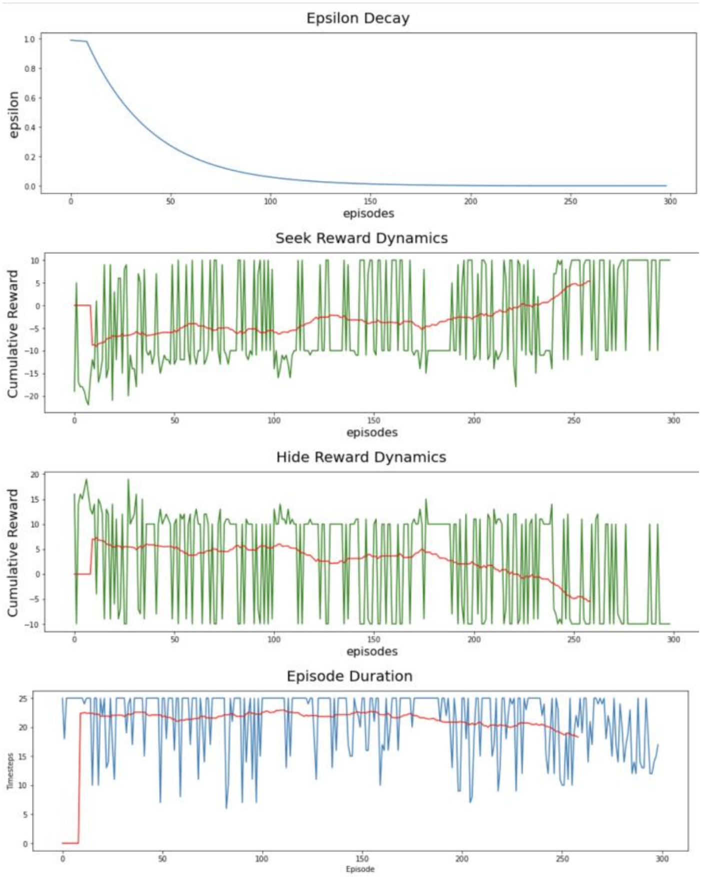
Number of distinct policies is fixed. There may be up to hundreds of policies (constrained by memory required to store weights).

Simple MARL Environment based on Grid world

- The environment is a simple gird environment, consisting of police and robber agents.
- The police catches the robber when it shares the same block as the robber.
- The robber has a speed that’s lower than the police. Safety is ensured by having a max timestep of 25 steps per episode.



Q-Learning on Gridworld

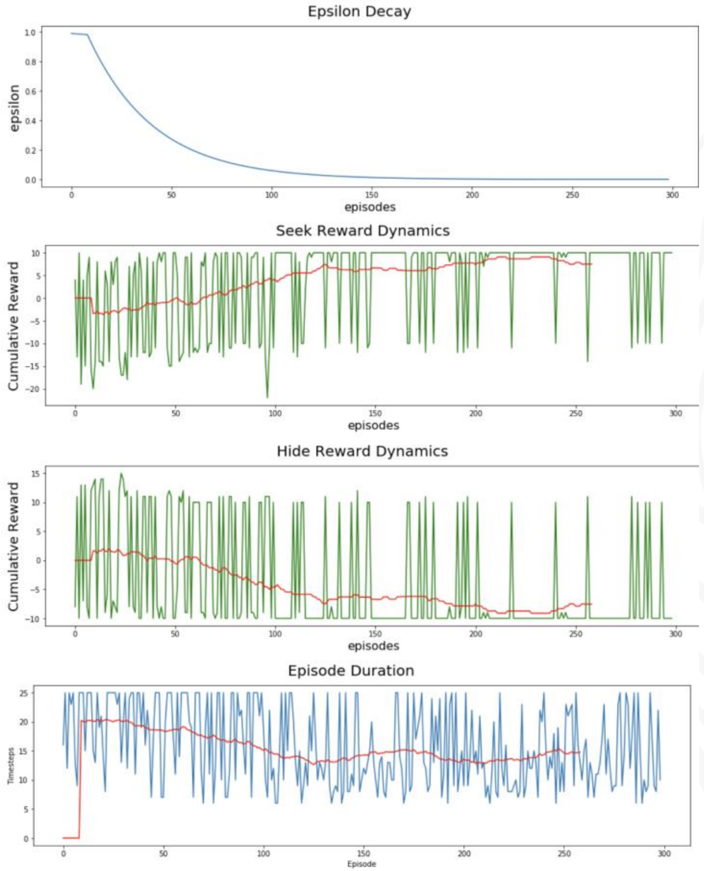


Epsilon Decay

Seek Reward Dynamics

Hide Reward Dynamics

Episode Duration

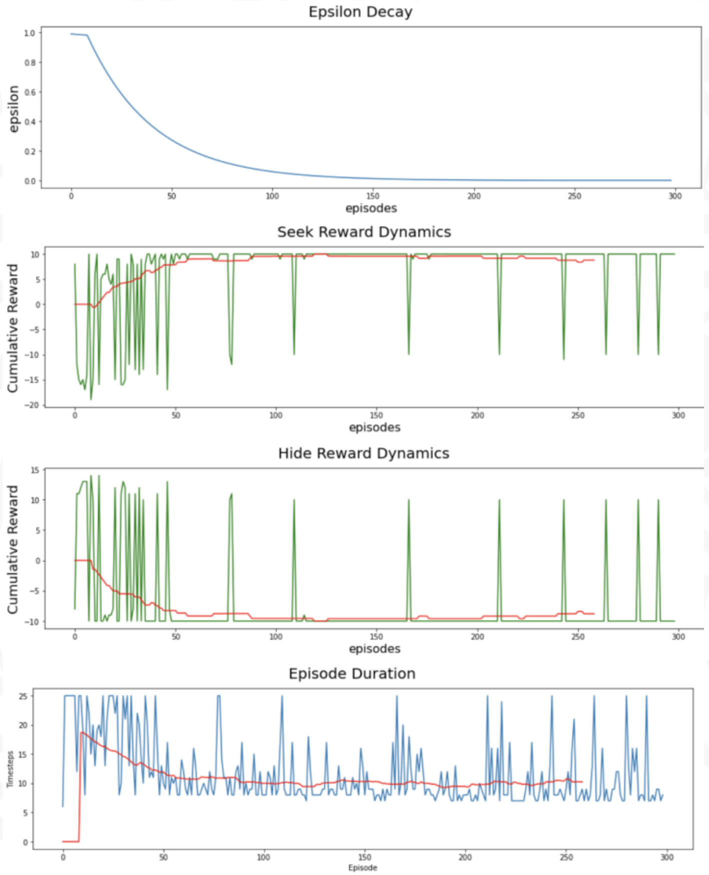


Epsilon Decay

Seek Reward Dynamics

Hide Reward Dynamics

Episode Duration



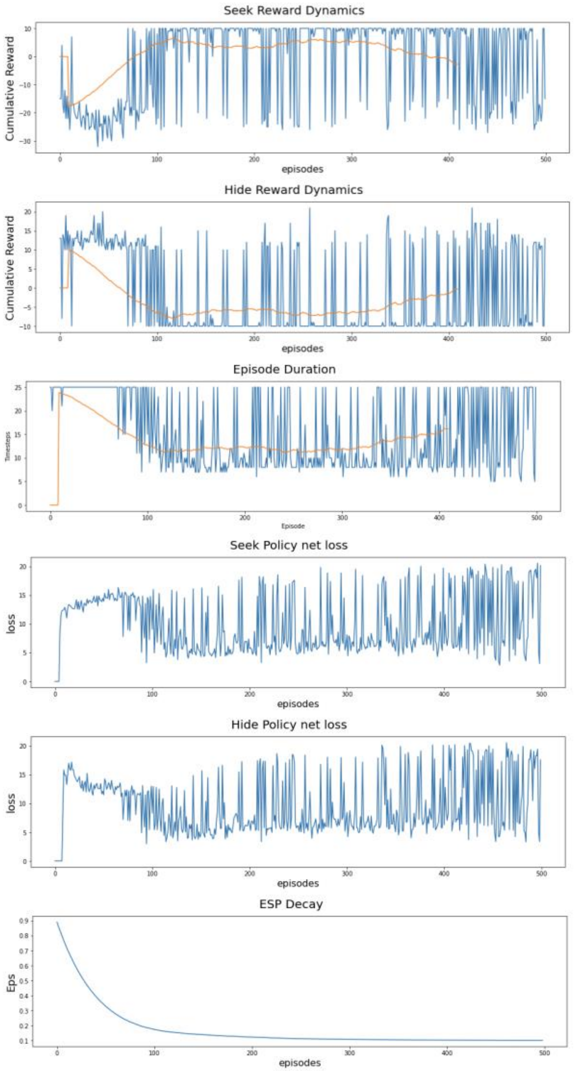
Epsilon Decay

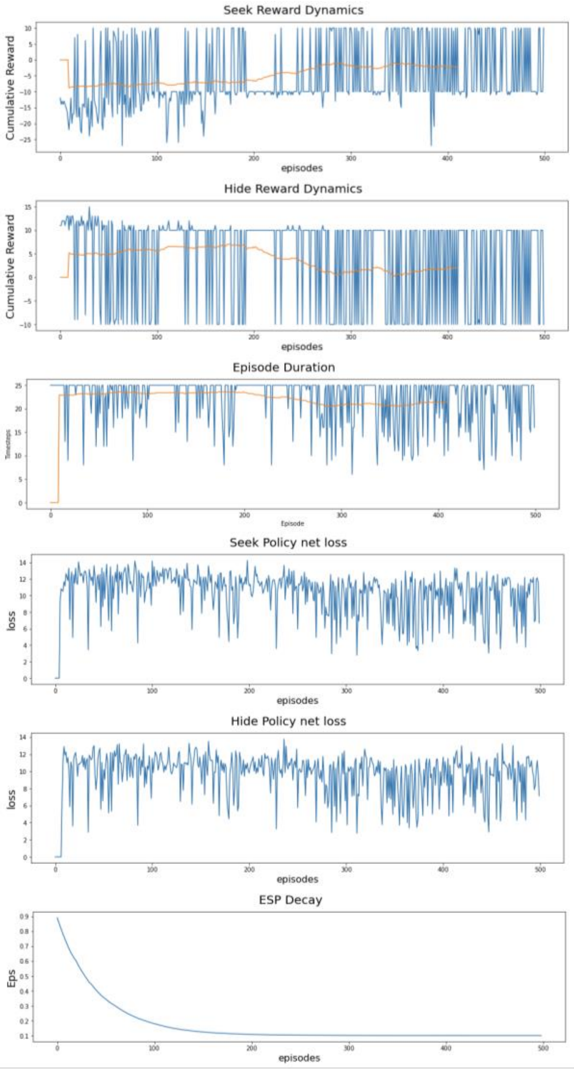
Seek Reward Dynamics

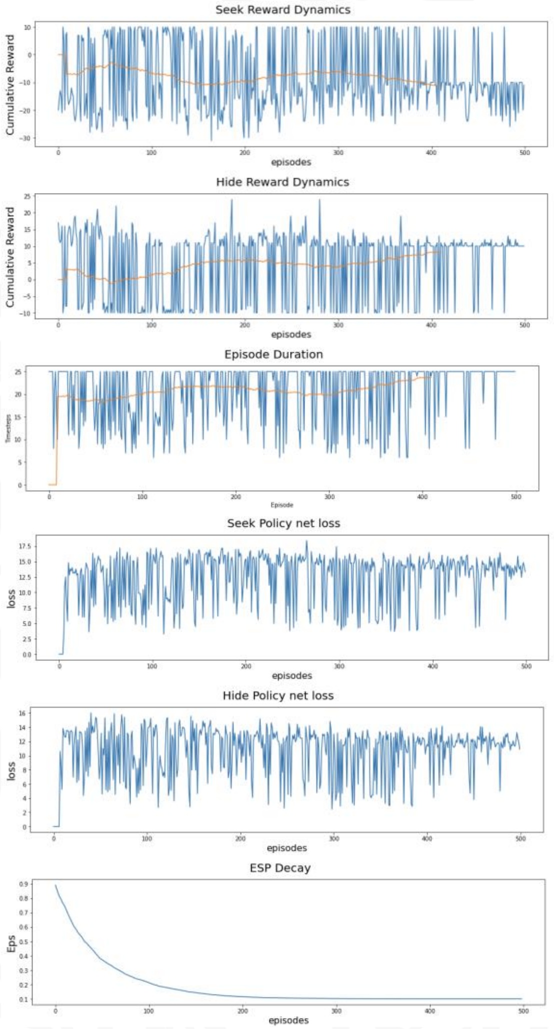
Hide Reward Dynamics

Episode Duration

DQN on Grid world

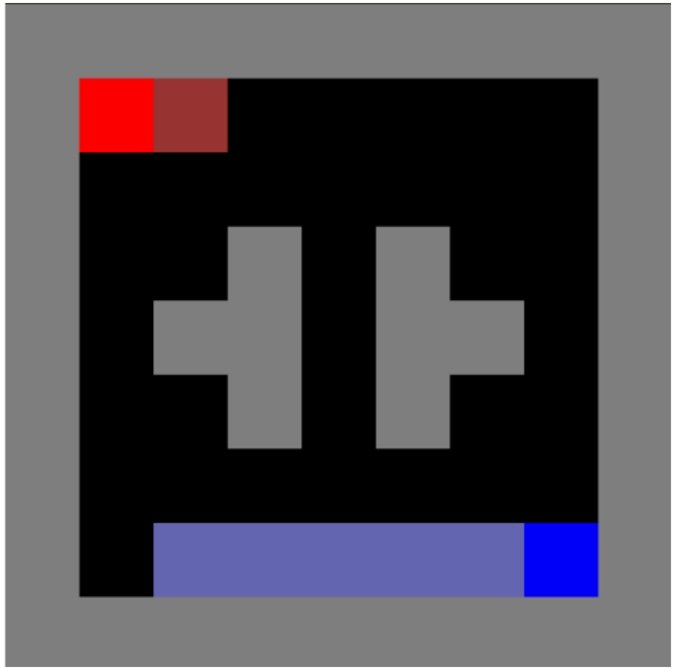


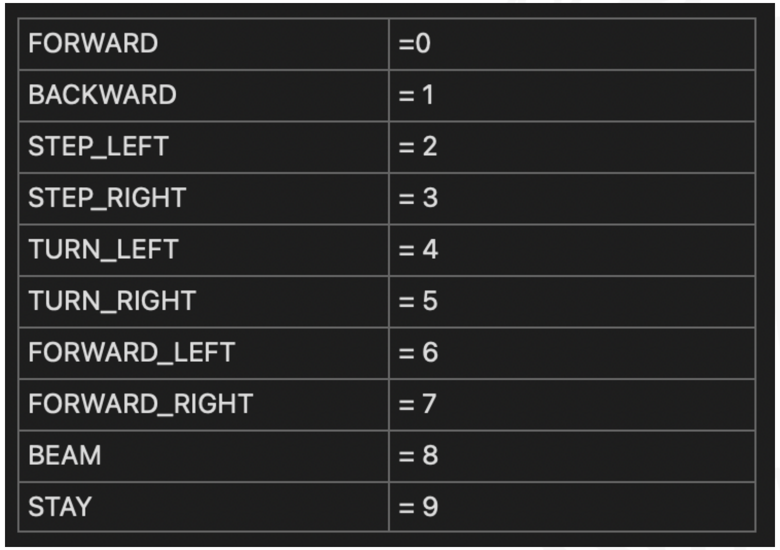




Complex MARL environment - LaserTag

- In this environment, two adversarial agents play a zero-sum game where each agent’s task is to zap the opponent to gain a reward. The agents have orientation as well, that is, the agents can zap only in 1 direction at a time. If an agent wins, it gets a reward 1 else 0 for every timestep.

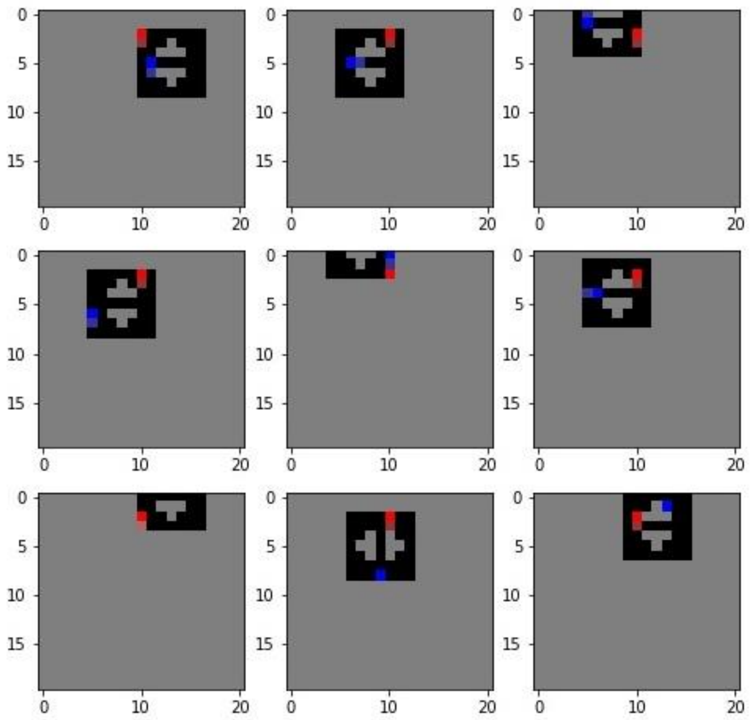




FORWARD	=0
BACKWARD	= 1
STEP_LEFT	= 2
STEP_RIGHT	= 3
TURN_LEFT	= 4
TURN_RIGHT	= 5
FORWARD_LEFT	= 6
FORWARD_RIGHT	= 7
BEAM	= 8
STAY	= 9

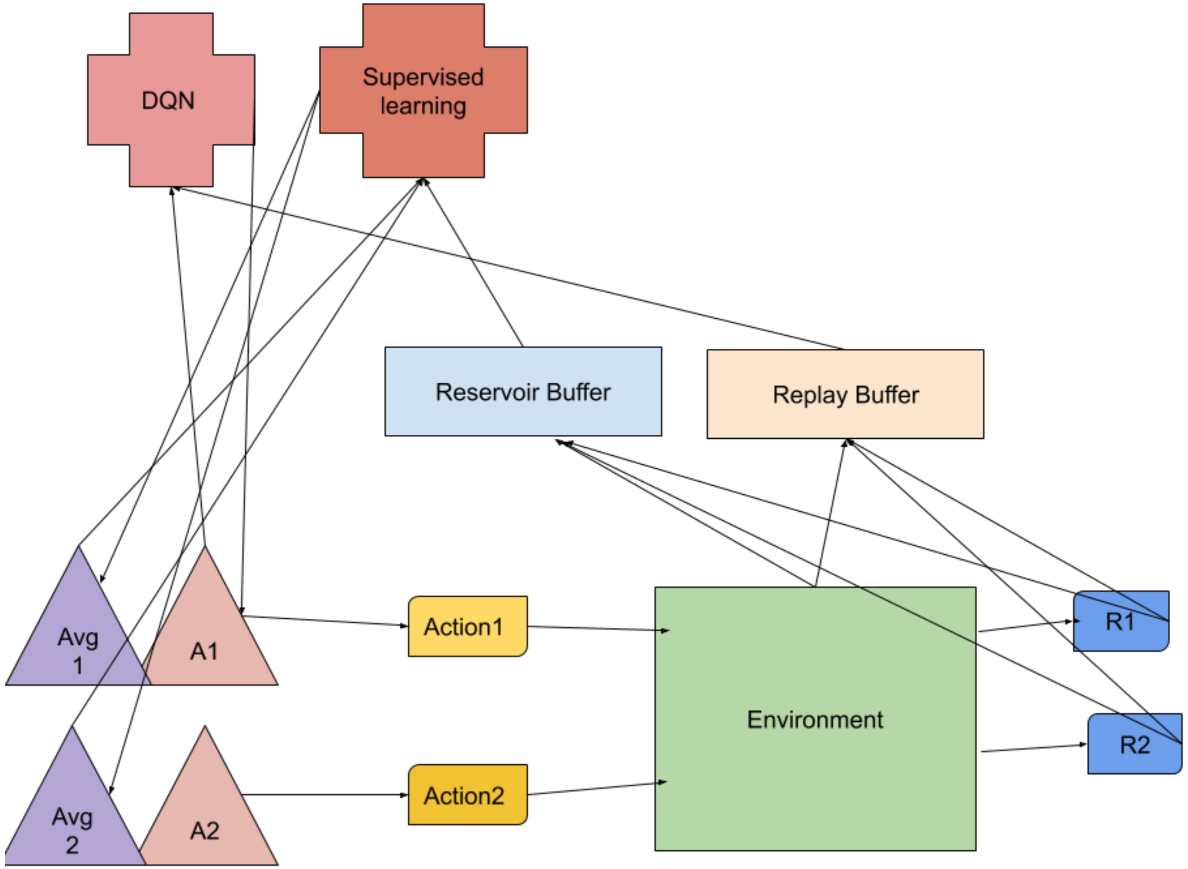
Partial visibility During training

- To ensure proper training, only a partial view of the environment is provided to both the agents during training. If the entire environment was visible, the game would be trivial.



Neural Fictitious Self Play

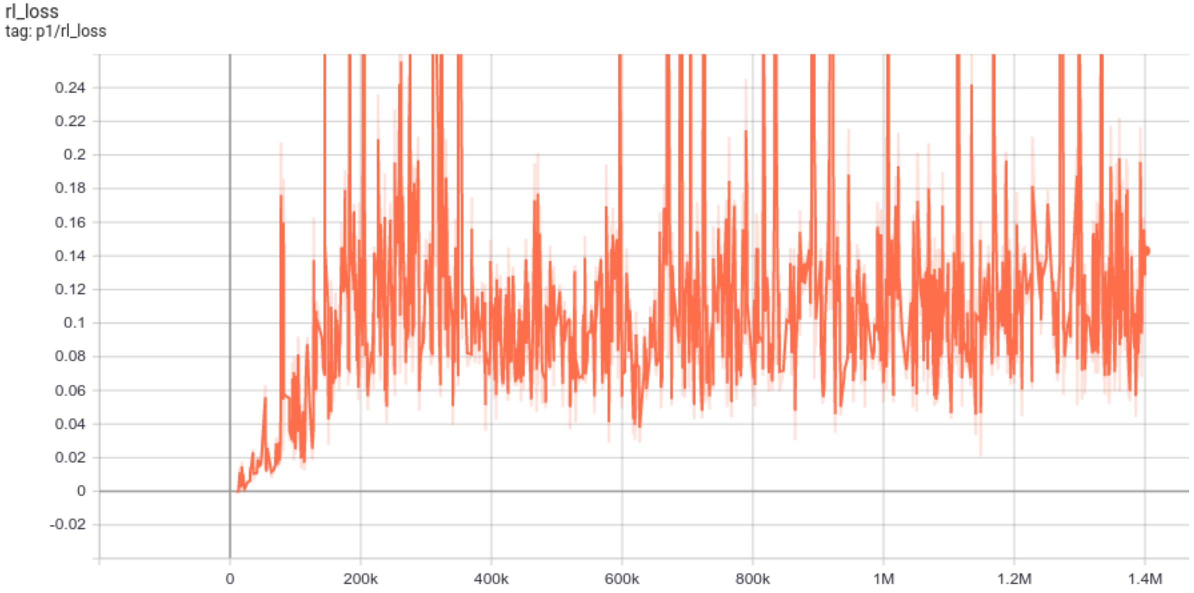
Deep Reinforcement Learning from Self-Play in Imperfect-Information Games
https://arxiv.org/pdf/1603.01121.pdf



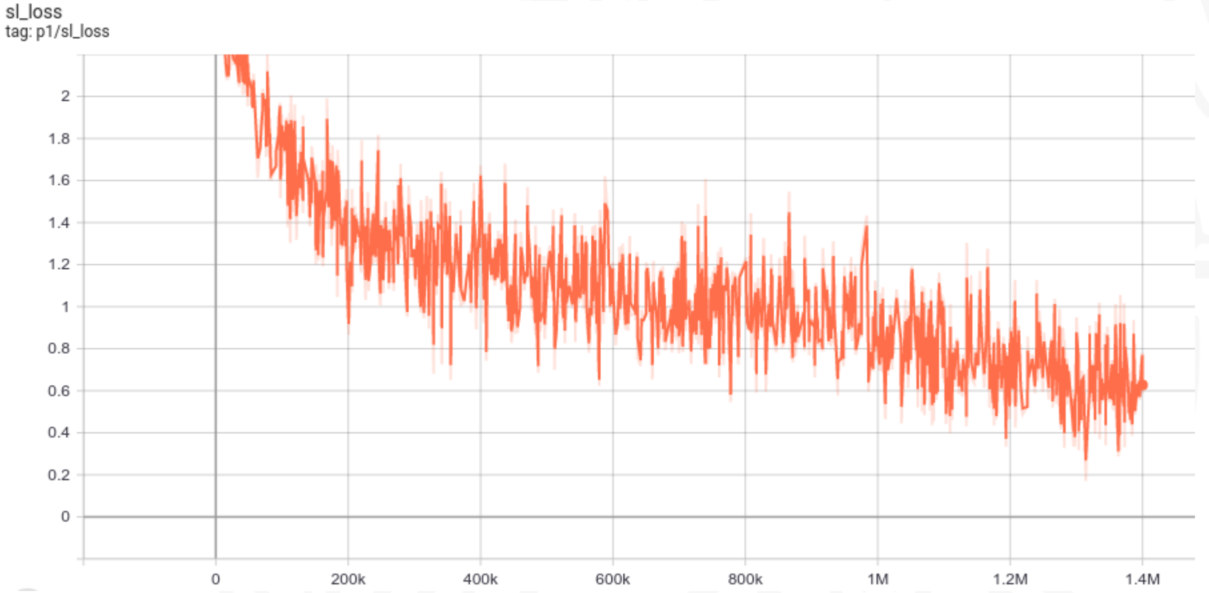
Training Procedure

- We have two networks, where one is trained in the RL framework while the other is trained as a supervised network.
- For the supervised network, the training data is stored in the reservoir buffer. It contains the actions that were performed using the RL network. The task of the supervised model is to predict what action was taken given the state. This can be viewed as an average policy.
- For the DQN network, we sample from replay buffer and predict the qvalues using the trainable model using current state and gather the qvalues for actions taken at current state. The next values are obtained using target network and next state. The max of next values + discounted reward is the expected qvalue. The difference between the qvalues from the trainable model and the expected qvalues is the loss.

Train Logs

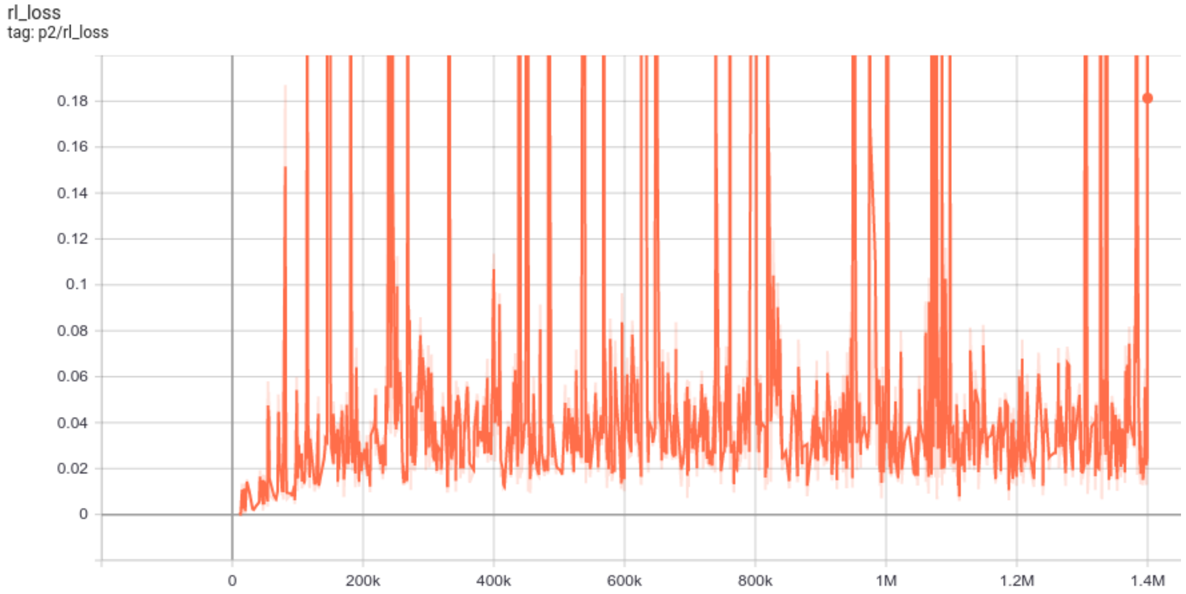


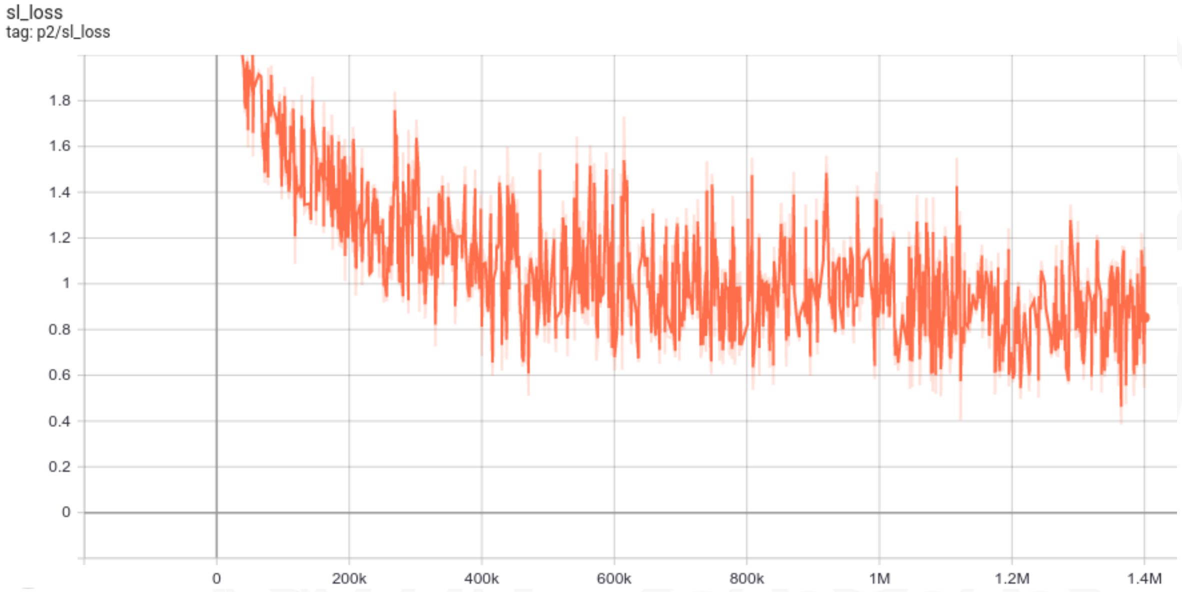
rl_loss
tag: p1/rl_loss



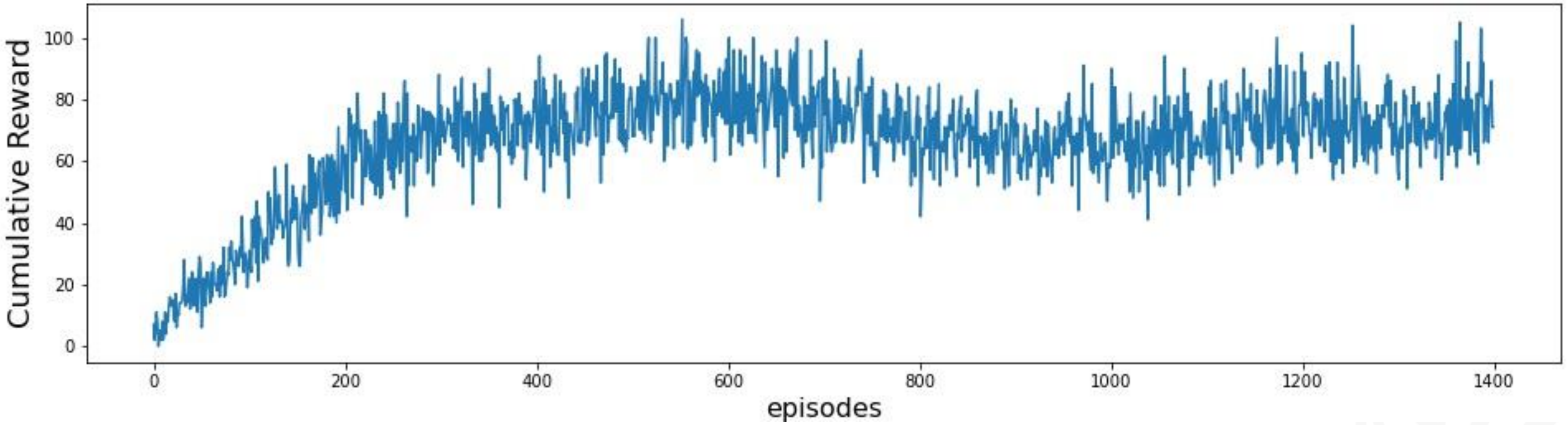
sl_loss
tag: p1/sl_loss

Train Logs

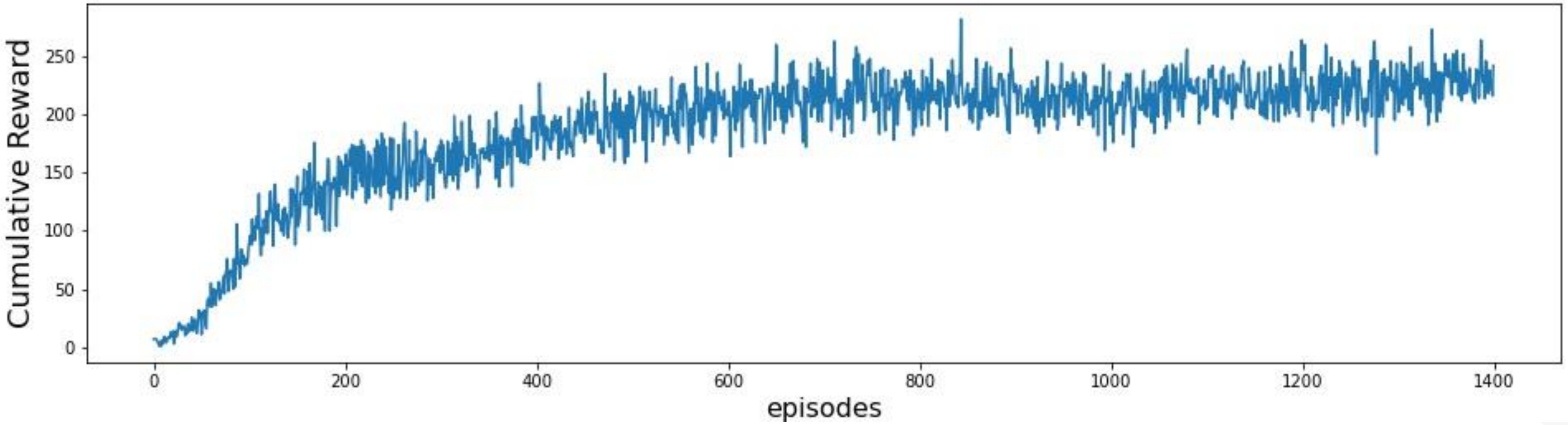

rl_loss
tag: p2/rl_loss
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
200k
400k
600k
800k
1M
1.2M
1.4M


sl_loss
tag: p2/sl_loss
1.8
1.6
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0
200k
400k
600k
800k
1M
1.2M
1.4M

Reward Dynamics

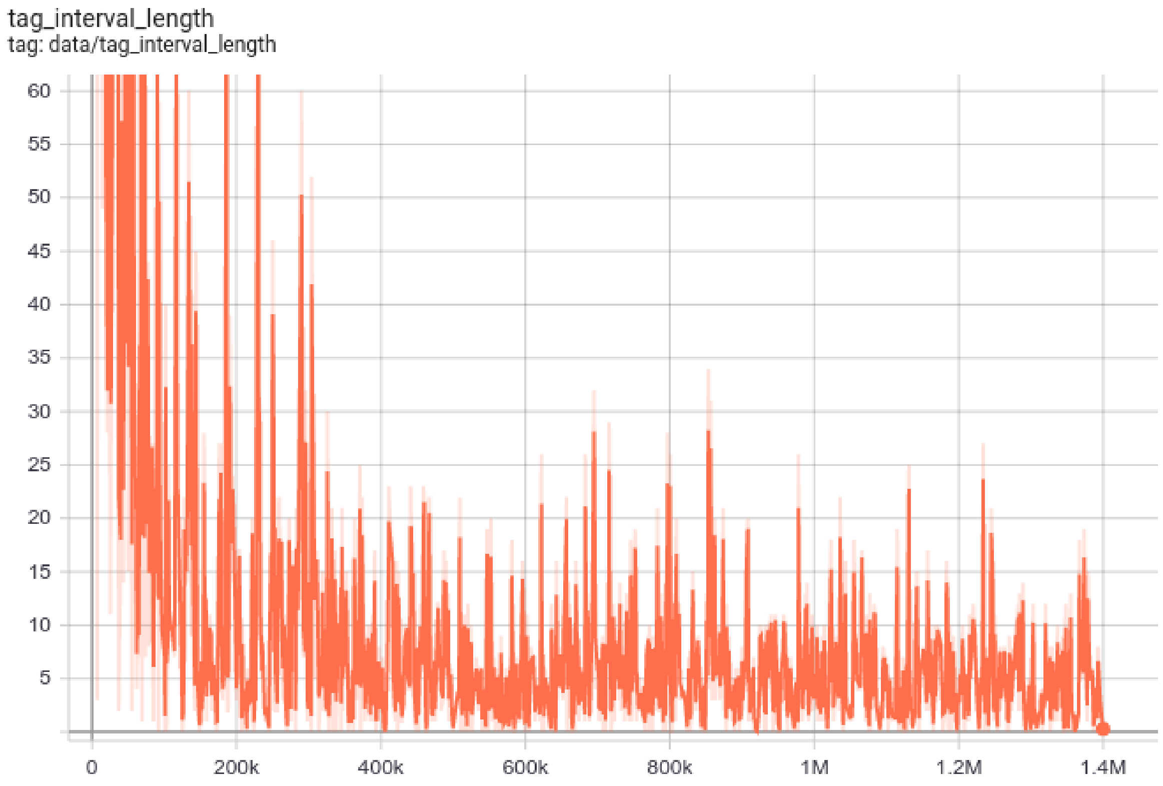


P2 Reward Dynamics



P1 Reward Dynamics

Episode Duration



Thank You