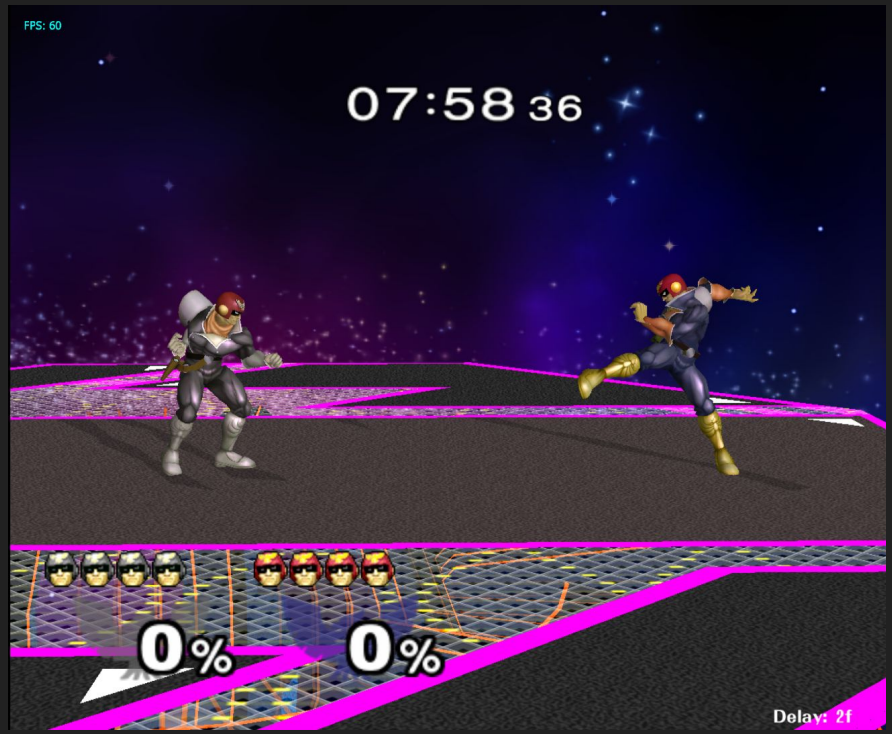
Is Transfer Learning Possible in Reinforcement Learning?

Scaling Difficulty on a Trained Agent in Super Smash

The What and the Why

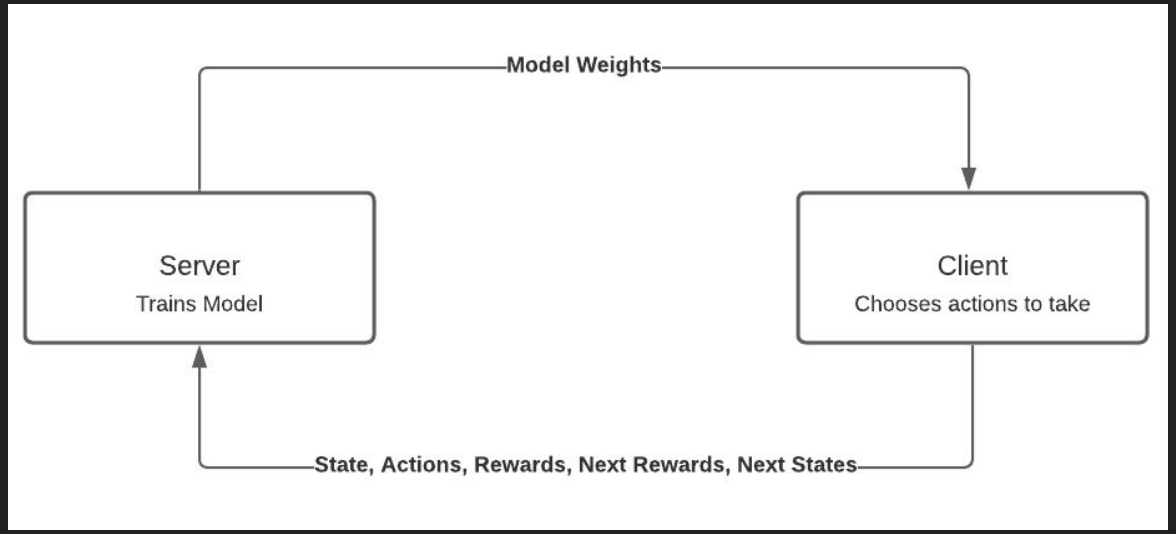
- Super Smash RL Agent
- Deep Q Learning in a complex environment
- Transfer learning applied to Reinforcement Learning
  - Increased difficulty against trained networks



Server-Client Model

- Django Server
  - Trains model

- Client
  - Runs the environment
  - Any Local Computer
  - Scalable



State Representation and Reward

- Reward Structure
  - +200 for kill, -200 for death, +1 per damage percent, -1 per damage percent
- State Representation
  - Knows for the 2 bots the following:
    - Lives, percentages, on the ground, off the stage, shield strength, x and y position, which direction it is facing, and its speeds in the x and y positions
  - 26 elements in total
  - Then adds previous move to the end of the state representation as recommended by Chris
    - 56 total elements in each state (30 different actions)

Learning Algorithm

- Deep Q Learning

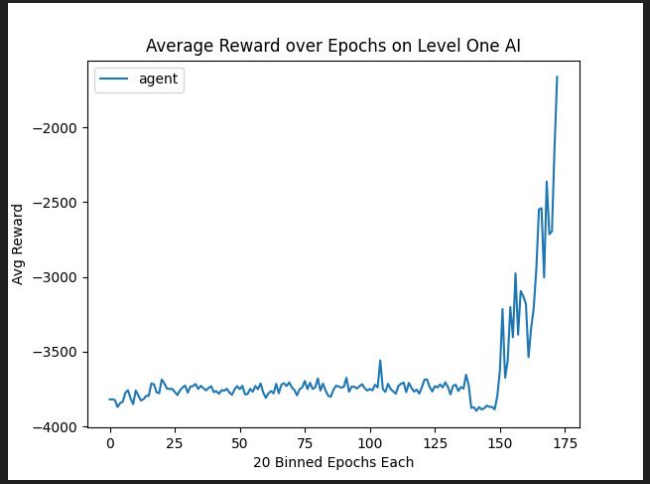
```
for state, action, reward, next_state, done in minibatch:
    target = reward

    if not done:
        next_state = np.array(next_state)
        next_state = next_state.reshape(1, -1)
        target = reward + float(agentHyperparameters.gamma) * np.max(model.predict(next_state))

    state = np.array(state)
    state = state.reshape(1, -1)
    target_f = model.predict(np.array(state))[0]
    target_f[action] = target
    target_f = target_f.reshape(1, -1)
    model.fit(state, target_f, epochs=1, verbose=1)
```

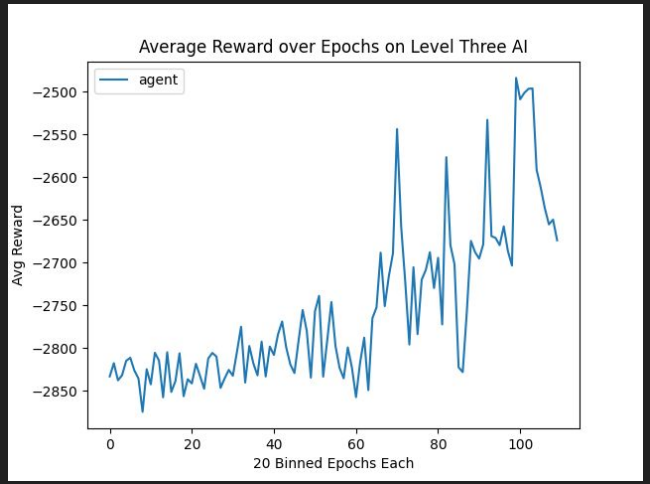
- Trains against Level One AI
  - Trains against more difficult AI afterwards
  - “Transfer Learning”

Results

- First Trained to beat a Level One AI
  - Required about 3500 epochs to begin to beat the AI

- Even after 2000 epochs didn't beat a Level Three AI using the weights from Level One
  - Ran again, with deeper network

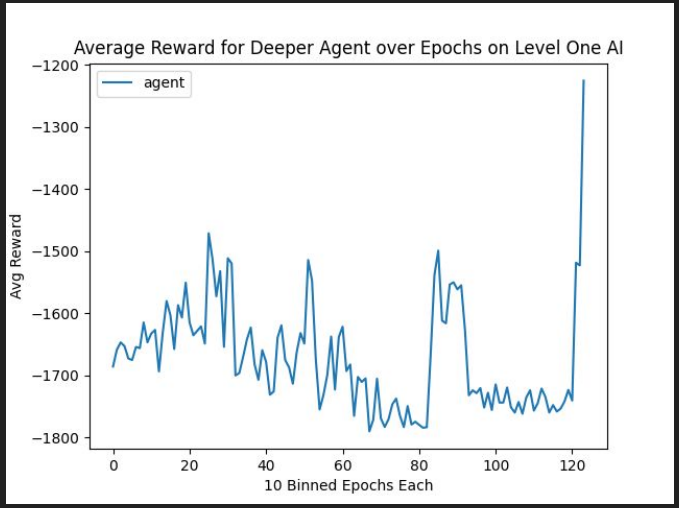


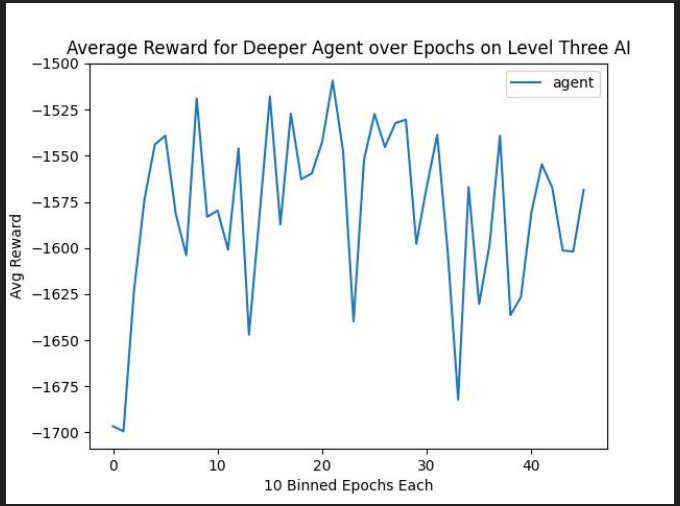


Results

Results

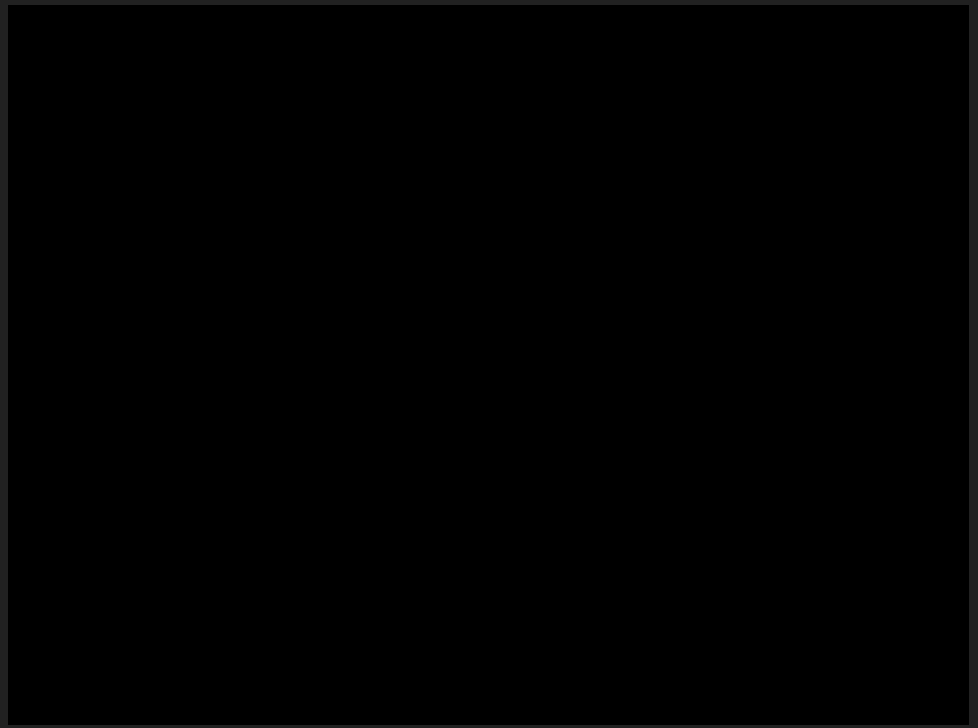
- Deeper Network Used
  - 5 Dense Layers instead of 3
- Had to train much quicker because of time constraints
  - Quick improvement, but not much from there





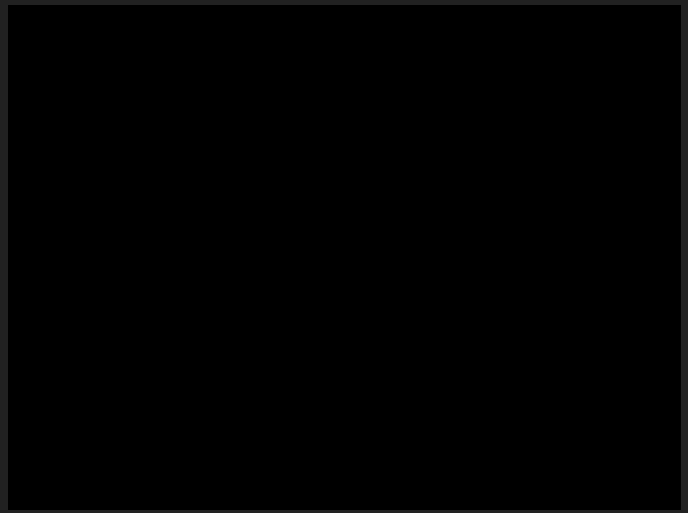
Results

- Does well then turns around and kills itself everytime after



Summary

- Hardly a difference in training agent to play against Level One/Level Three bot
    - At least, so far
- Does train enough to kill sometimes
    - Tends to kill itself a lot
- Uses same move over and over
    - Good for level one bot, level three easily beat it
- Potentially, slower scaling of difficulty
    - Level one to Level three is a big jump
    - Train against level two first might yield better results



Limitations and Future Work

- Takes a really long time to train
  - Complicated state space and large action space
- Reward for particular moves used
- Watch games of professionals and learn from them for a baseline
- Implement an Actor Critic network or PPO instead of DeepQ for faster training
- Scale Clients, have more running environments
  - Parallel Training