OpenPrison

Can Deep Learning beat the Algorithms?

Daniel Gonzalez, Nathan McCutchen, Luke Rowe, Nicholas Tan, Christine Widden

CAL POLY
College of Engineering

Background

- **Prisoner's Dilemma** - A thought experiment where two participants choose to cooperate for mutual gain or betray the other for personal gain.
- **Axelrod's Tournament** - Numerous game theorists competed to devise the best Iterated Prisoner's Dilemma (IPD) strategy
  - The original tournament featured 14 different strategies. The winner was *Tit-for-Tat*, one that simply copied its opponent's last move [3].

Goal, Motivation, Requirements

- Since our class had an AI agent based theme, our group wanted to explore the prisoner's dilemma experiment, and evaluate the emergent behaviors of AI agents trained against different logic actors.
- **Our main goal** for this project was to train a set of DRL agents against varying logic-based opponent pools to observe the impact of opponent strategies on emergent behaviors

Requirements

- Recurrent DRL algorithm to train our agents
- Environment to train/run 2-agent games
- Existing logic-based algorithms

System & Implementation

The Model

- Learning agent utilizing a Recurrent PPO model
- PPO (Proximal Policy Optimization) is a deep reinforcement learning (DRL) algorithm that attempts to maximize reward by searching for optimal behavior, taking small steps at a time [4].
- We use the *Mlp Lstm Policy* which uses multilayer perceptron (MLP) layers and Long Short-Term Memory (LSTM) layers. It's suitable when the game state representation requires sequential processing (e.g., past actions or observations) [6]

The Environment

- We used Stable Baselines 3 and OpenAI's Gymnasium Python packages to implement our learning agent model and training environment
- We implemented an abstract for every agent defining how the run script interacts with them
- The run script accepts a set of two participant agents, logic-based or DRL-based, and runs an IPD game, returning the results of the match

Command Line Interface

Training Interface

```
-------------------------------------
| rollout/                |         |
|    ep_len_mean          | 100     |
|    ep_rew_mean          | 142     |
| time/                   |         |
|    fps                  | 293     |
|    iterations           | 1465    |
|    time_elapsed         | 639     |
|    total_timesteps      | 187520  |
| train/                  |         |
|    approx_kl            | 0.0     |
|    clip_fraction        | 0       |
|    clip_range           | 0.2     |
|    entropy_loss         | -0.000142 |
|    explained_variance   | 0       |
|    learning_rate        | 0.0003  |
|    loss                 | 110     |
|    n_updates            | 14640   |
|    policy_gradient_loss | -1.12e-08 |
|    value_loss           | 220     |
-------------------------------------
  6% ──
```

In-Game Interface

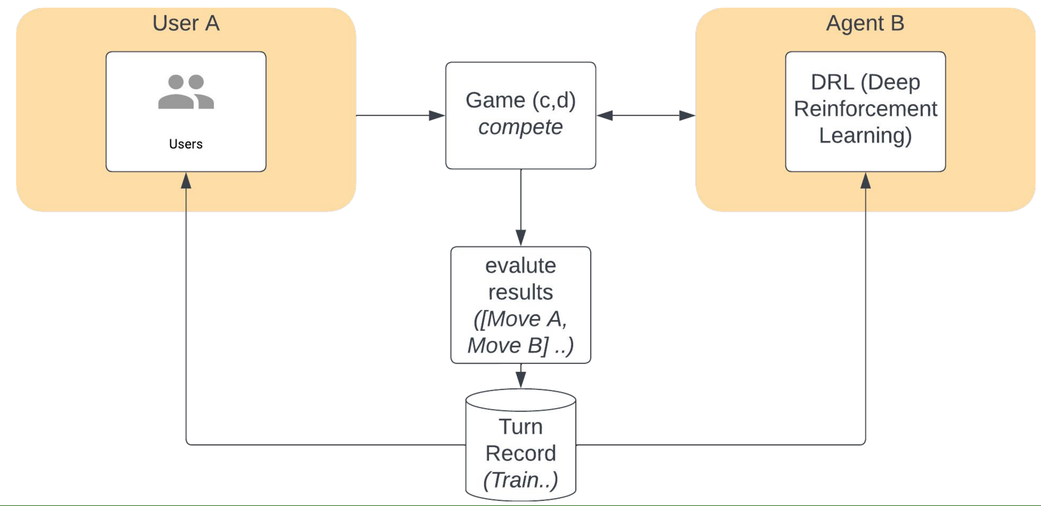
```
----------------Round 3-----------------

Scores:
        Human: 5
        Fred: 2

INPUT: Human
        Cooperate or Defect? (c/d): c

Results:
        Human: cooperate (0)
        Fred: defect (3)
```

System Diagram



References

1. https://sb3-contrib.readthedocs.io/en/master/modules/ppo_recurrent.html
2. https://colab.research.google.com/github/Stable-Baselines-Team/rl-colab-notebooks/blob/sb3/stable_baselines_getting_started.ipynb
3. https://axelrod.readthedocs.io/en/fix-documentation/reference/overview_of_strategies.html
4. https://ppo-details.cleanrl.dev//2021/11/05/ppo-implementation-details/
5. https://gymnasium.farama.org/index.html
6. https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

Results & Analysis

Results

- We were pleased to find that the trained models had **differing resultant behavior** depending on the distribution of the opponent pools
- We found the **most common behavior** to be similar to our *Always Defect* logic-based agent
  - This was common when we included a *Random* agent in the opponent pool
- The **second-most common behavior** DRL agents seemed to lean towards was similar to *Tit-for-Tat*
  - An opponent pool of *Tit-for-Tat*, *Generous Tit-for-Tat*, *Always Defect*, *Always Cooperate*, and *Grim Trigger* seemed to consistently produce this behavior

Analysis

- We attribute the nature of our emergent behaviors to the fact that our reward function only accounts for the results of a single iteration
- This incentivizes a DRL agent to maximize it's reward for an individual match, but an overall score across all matchups (with differing opponents) is not accurately represented
- Thus it would become necessary to introduce a concept of cross-episode performance
- This seems difficult to achieve given the current state of our system, and would require future research to successfully implement

Future Work

- Implementing a version of our system that accounts for cross-episode performance may result in more interesting emergent behaviors
- Adding to the selection of logic-based agents to train against may diversify behaviors
- We imagine an alternate version of IPD where agents can communicate intentions before each turn; emergent behavior of DRL models trained with pre-turn communication could be an interesting avenue for future research

GitHub:
N8WM/open-prison