Course: CS 5100 - Foundations of AI

Project Team: Beini Wang, Frank Anderson, Jack Turner, Rahul Suresh

Project Proposal

Overview

Our team will be training an AI agent to be a master Uno! player. Uno! is a classic card game where players compete against each other to be the first to run out of their cards.

Cards are one of four colors—red, green, blue, and yellow— and have numbers zero through nine. In addition, there are cards with special abilities, such as skip to skip the next player, reverse to change the direction of play, draw two to force other players to add cards to their deck, and wild which lets players change the color being played.

To play, players go one-by-one playing a card from their hand with a matching color or number to the card currently in play. If a player does not have a valid card in their hand, they draw a card from the draw pile and play moves on to the next player. Players continue until one player has run out of their cards and is deemed the winner!

Motivations

Our team is excited about this project topic in particular because we all share an affinity for board games. We chose Uno! in particular because it's a popular game so most members of the class will be able to understand the presentation without being confused too much by the rules of the game. Additionally, it's an intriguing game since it appears mostly random on the surface, but we believe there is strategy to be uncovered. We are curious if the agent we develop can find a strategy we have not considered which will let us dominate future game nights played with friends.

Existing Approaches

In our research, we found two potential approaches for developing an AI agent to master Uno!, though there are undoubtedly many other methods.

Approach 1: Reinforcement Learning (GitHub, Article)

In this example, Reinforcement Learning is used to improve the agent. To model the game, a 16-digit state identification is modeled to significantly reduce the size of the state space. Then two reinforcement learning techniques (Monte-Carlo, Q-Learning) are applied over 100,000 simulated games to improve the agent. A key finding was 32% of turns had only one playable card, so there is only a limited set of ways for strategy to win out.

Approach 2: Genetic Algorithm (GitHub)

In this example, a Genetic Algorithm is applied to find an optimal configuration for the agent. Points are awarded to each card for their value when played, and then the configuration was adjusted over many generations to find the optimal point distribution. While this may have produced a working result for the programmer, we are concerned that assigning values to the cards in this way is relatively arbitrary, and the value of individual cards will be highly situational and change throughout the game– something this approach does not accommodate.

Supporting: Existing Uno! Python implementation (GitHub)

We found this implementation of Uno! which we plan to build our agent for. The implementation allows us to replace one, or multiple, players with AI agents or human players, so we can spar against our Agent and the Agents can spar against one another.

References

Approach 1

- https://github.com/bernhard-pfann/uno-card-game-rl
- https://medium.com/towards-data-science/tackling-uno-card-game-with-reinforcement-learning-fad2fc19355c

Approach 2

- https://github.com/AwSkies/uno

Supporting

- https://github.com/bennuttall/uno