An Artificial Intelligence for a Crazy Eights game

Matias Castillo, Benjamin Goeing, Jesper Westell

PROJECT OBJECTIVE

- Our objective is to develop a Game-playing AI that can interact and play a game of Crazy Eights against a human player. We have chosen this game because it has the challenge of possessing a huge state space, that each player can only partially observe
- We implemented a popular version of the game using a state-based model, created intelligent agents using different learning techniques, and compared how they performed against a Baseline and an Oracle
- For CS 229, we have developed a computer vision algorithm that recognizes the state of a game based on a photo of the table using a CNN. This can be combined with our AI agent to create a new interactive Crazy Eights game

THE GAME: RULES AND DESIGN

- Each player starts with 7 cards in his/her hand. The objective of the game is to get rid of all your cards before your opponent does
- One card from the remaining deck gets turned face up in the middle of the table. Players then take alternating turns to play a card from their hands on top of the card on the table until one player finishes
- A player is only allowed to play a card from his/her hand, if it matches the suit or rank of the card lying on the table. Otherwise, he/she needs to draw a card from the deck and add it to their hand.
- Eights are allowed to be played at any time, and serve as a kind of "joker" in the game

AGENTS AND MODELS

We tested 4 different agents against one another:

- **Baseline Agent:** Randomly chooses an action among the valid actions for a given state
- **Oracle Agent:** A modification of an existing Crazy Eights agent found at https://repl.it/@christianrasmussen/cardgames-crazy-eights
- **Minimax Agent:** An agent using alpha-beta pruning and simple evaluation function returning the negative number of cards in hand
- **RL Agent:** Evaluates states using the feature vector and parameters described in the Reinforcement Learning section

REINFORCEMENT LEARNING

- A key challenge in this projects is the possible number of states, as there are over 130 million initial start states alone
- We defined a feature vector, that extracts relevant information from the game state, thereby greatly reducing the number of states to make reinforcement learning possible

$$\phi = \begin{cases} \#\text{ cards in hand} \\ \#\text{ opponent cards} \\ \#\text{ eights in hand} \\ \#\text{ cards in deck} \\ \#\text{ cards with same rank} \\ \#\text{ cards with same suit} \\ \text{probability distribution of unkown cards} \end{cases}$$

- Training examples are generated using Monte Carlo simulations

$$\mathcal{D}_{train} = \{(x^{(i)}, y^{(i)}) : x^{(i)} \text{ randomly generated state },$$

$$y^{(i)} = \frac{1}{N}\sum_{j=1}^{N} E\{X_{i,j}\}\}$$

$$X_{i,j} = \begin{cases} 0, & \text{Oracle wins game } G_{i,j} \\ 1, & \text{Minimax agent wins game } G_{i,j} \end{cases}$$

- Value of a state learned through repeated simulations from that state and averaging the amount of won games
- We then used linear regression to learn the parameters for the feature vector to be used by the RL Agent

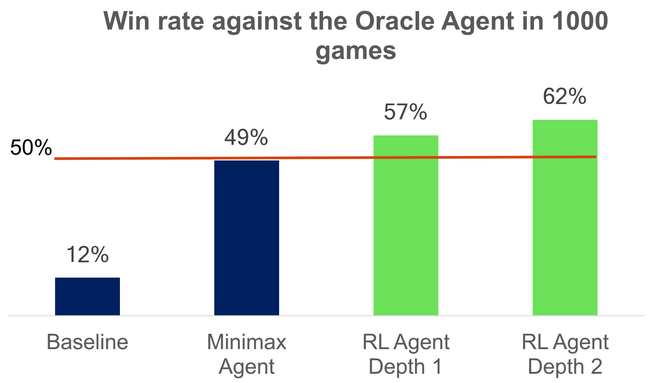
PRUNING THE SEARCH TREE

- Many possible actions an opponent could take, result in a long computational time to perform search with high depth
- Using model-based Monte Carlo, we approximate the state-action transition probabilities $P(A = a, S = s)$, $P(S = s)$
- Then, using a probability distribution, we prune and only consider the most probable actions:

$$P(A = a \mid S = s) = \frac{P(A = a, S = s)}{P(S = s)}$$

- This allows us to use a reinforcement learning agent with depth 2

RESULTS AND CONCLUSION



- The minimax agent already performs quite well against the oracle, which shows that even a simple agent is a lot better than the Baseline
- The RL Agent shows improvement and beats the oracle on average, which shows the success of our approach
- Increasing the depth, that is possible thanks to additional pruning, shows even more improvement
- As there is a significant amount of chance involved in this game, we feel that even with more computational power, our results would only improve marginally