CS229: Reinforcement Learning Applied to a Game of Deceit

Hana Lee – Stanford University, M.S. in C.S.

Problem

Question: How useful is RL as a tool for training deceit?
Application: Skull, a simple betting/bluffing game played by 4-6 players
Goal: Train a reinforcement learning agent to win at Skull by making convincing bluffs

Fig 1: Skull and flower tiles from the game of Skull

Method

Method: Model a simplified version of 2-player Skull as a Markov Decision Process (MDP) and find an optimal policy by policy iteration

$$\pi(s) := \arg\max_{a \in A} \sum_{s'} P_{sa}(s')V(s')$$

Game: Both players place at least 1 tile, then bet in increments of 1 until someone passes. If betting player can flip over a # of flowers equal to their bet, they win. If they flip over a skull, they lose.
State: RL agent's stack, opponent's stack, current bet amount
Actions: Add Flower, Add Skull, Bet, Pass

MDP

$\mathbf{P_{sa}}$ = hardcoded approximation of real play
$\mathbf{R_{sas'}}$ = reward of 1 for victory, -1 for loss
ε = random walk parameter
Bluff rate = % of optimal actions which are bluffs

ε	Iterations to converge	Bluff Rate
0	7	0.22
0.05	19	0.23
0.1	86	0.23
0.15	311	0.23
0.2	2124	0.24

Fig 2: Results from MDP with no uncertainty

Results:
- Bluff rate varies only slightly as random walk parameter changes.
- Introducing a reward for bluffing doesn't change the rate at all.

Evaluation

And now for the fun part… playing against the AI! Each time the AI makes a bet, I decide whether I believe it's bluffing and the game records my decision.
- **Only best actions**: I win most games, and it's easy to tell when the AI is bluffing.
- **70% best, 30% runner-up**: It's an even split, and I guess wrong more often.

POMDP

Obstacle: In a real game, we don't know which tile in our opponent's stack is the skull; exact state is uncertain.
Solution: Maintain uniform belief distribution over all possible skull locations.

ε	Iterations to converge	Bluff Rate
0	6	0.12
0.05	6	0.12
0.1	6	0.12
0.15	27	0.12
0.2	48	0.12

Fig 3: Results from POMDP with state uncertainty

Results:
- Much lower bluff rate with uncertainty.
- Still no change as random walk or reward varies.

Future Work:
- Use social learning to train several RL agents.

	Only best	70%/30%
AI Win Rate	20%	50%
Correct Guesses	89.5%	57.1%
False Positives	10.5%	28.6%
False Negatives	0%	14.3%

Fig 4: Results from playing against the AI and trying to guess when it's bluffing