

Deep CFR for HUNL

Nima Chitsazan
Foad Khoshouei
Ridhi Mahajan

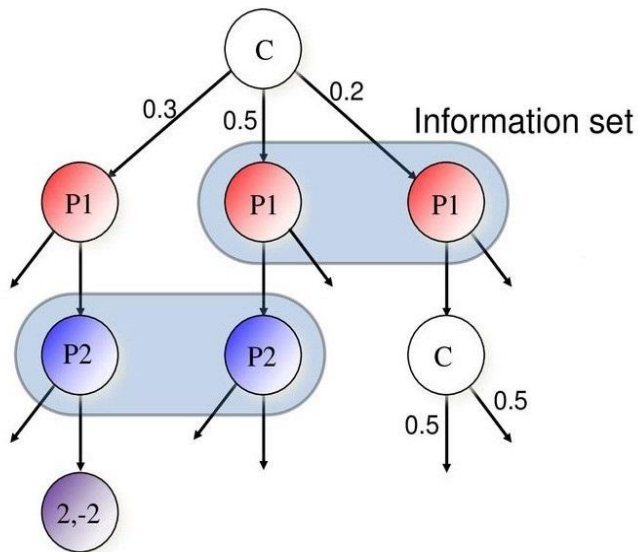
Imperfect Information Games

- Information available to the agent is limited (Meyerson 1997)
- Applications: Negotiation, Managing Online Auction, Navigating Traffic
- Poker has been widely used for testing
- Solving the game is to find Nash Equilibrium
- No improvement by deviating from the equilibrium



Related Work

- Counter-Factual Regret Minimization (CFR) (Zinkevich 2007)
- CFR+, Discounted CFR, Monte Carlo CFR (MCCFR)
- Deep CFR, Single Deep CFR
- Deep Reinforcement Learning- Neural Fictitious Self Play (NSFP) (Heinrich 2016)
- Deep Stack (Bowling 2017)
- Libratus (Brown 2018)
- Pluribus (Brown 2019)



CFR, Deep CFR Methods

- CFR finds the Nash Equilibrium by minimizing Counter Factual Regret
- By iteratively updating the weights, it converges to Nash
- CFR is computationally expensive
- Deep CFR uses Neural Network to approximate the Regret

Algorithm 1 Deep Counterfactual Regret Minimization

function DEEPCFR

Initialize each player's advantage network $V(I, a|\theta_p)$ with parameters θ_p so that it returns 0 for all inputs.

Initialize reservoir-sampled advantage memories $\mathcal{M}_{V,1}, \mathcal{M}_{V,2}$ and strategy memory \mathcal{M}_Π .

for CFR iteration $t = 1$ to T **do**

for each player p **do**

for traversal $k = 1$ to K **do**

 TRAVERSE($\emptyset, p, \theta_1, \theta_2, \mathcal{M}_{V,p}, \mathcal{M}_\Pi$) \triangleright Collect data from a game traversal with external sampling

 Train θ_p from scratch on loss $\mathcal{L}(\theta_p) = \mathbb{E}_{(I, t', \tilde{r}^{t'}) \sim \mathcal{M}_{V,p}} \left[t' \sum_a \left(\tilde{r}^{t'}(a) - V(I, a|\theta_p) \right)^2 \right]$

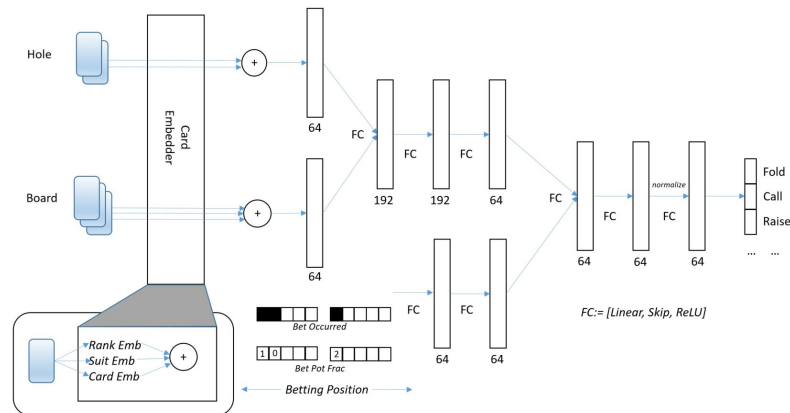
 Train θ_Π on loss $\mathcal{L}(\theta_\Pi) = \mathbb{E}_{(I, t', \sigma^{t'}) \sim \mathcal{M}_\Pi} \left[t' \sum_a \left(\sigma^{t'}(a) - \Pi(I, a|\theta_\Pi) \right)^2 \right]$

return θ_Π

Source: Deep Counterfactual Regret Minimization
(Brown et al. 2019)

Network Architecture

- Private cards and the community cards are embedded separately into two vectors with dimension 64.
- Followed by three dense layers with size 192 (Card Layers).
- Two dense layers of size 64 (Betting Layers). The information from the last betting layer and the last card layer are combined and conveyed through three (3) layers of size 64 (Comb- Layers).



Source: Deep Counterfactual Regret Minimization
(Brown et al. 2019)

Test Problem

Leduc Poker:

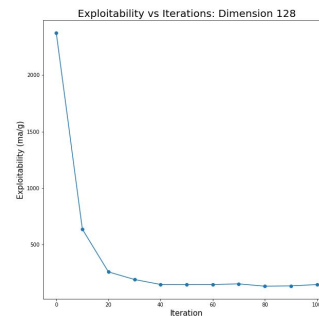
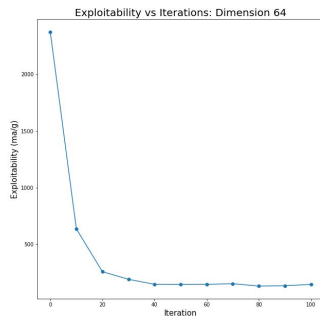
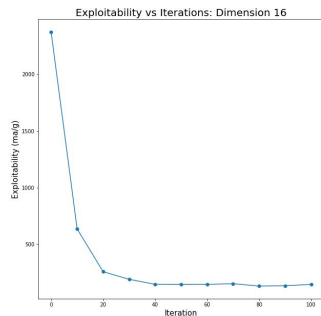
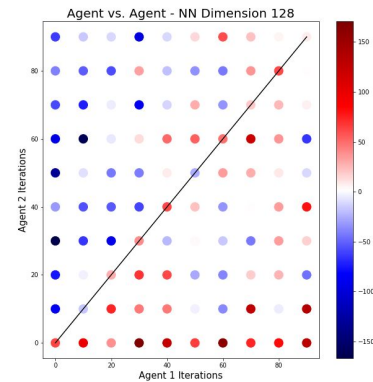
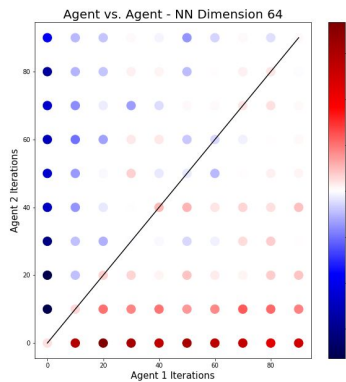
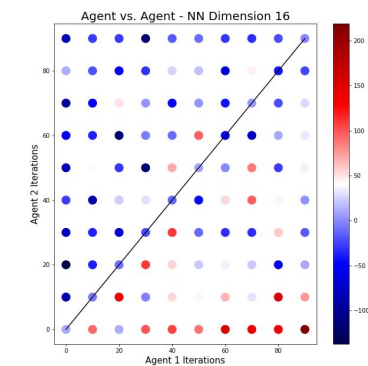
- 2 Suits, 3 Ranks of Each
- 1 Private Card
- 1 Community Card

Heads-UP No Limit Hold'em (HUNL):

- 52 Cards
- 2 Private cards
- 5 Community Cards
- 4 Rounds of betting
- 10^{161} Game States



Results For Leduc

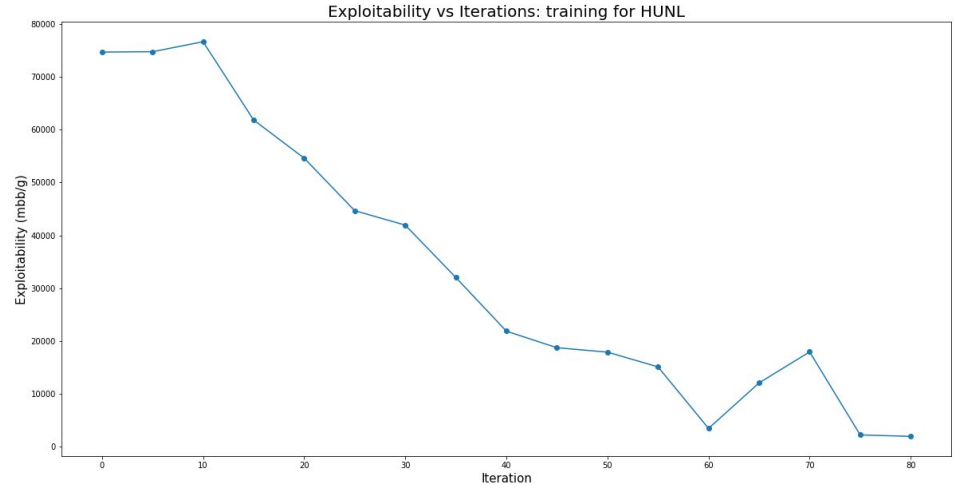


Methods to Implement Deep-CFR for HUNL

Issue	Solution	Implemented?
Large number of actions -- High branching factor	Action Abstraction	✓
High running time	Adaptive Traversals Per Iteration	✓
Costly function to compute	Local Best Response (LBR)	✓
Training the network from scratch at each iteration is slow	Warm Start -- Use weights from previous iteration	✓
Large number of states	Card Abstraction	✗

Results for HUNL

- Agent Trained on the game of HUNL
- Exploitability vs Iteration shown for HUNL
- Trained agent can play against humans reasonably



Results for HUNL

```
mahajan14@Tabletop: ~/Deep-CFR
Actions:
0 Fold
1 Call
2 Raise 50.0 % of the pot
3 Raise 75.0 % of the pot
4 Raise 100.0 % of the pot
5 Raise 200.0 % of the pot
6 Raise 1000000.0 % of the pot

*****
+ GAME START
*****

preflop - 0 acts
Board:
Last Action: player_None: None
Player_0:stack: 19950 current_bet: 50 side_pot_rank: -1 hand: 5s, 4h, | Side_pot0: 0 None | Main_pot: 0
Player_1:stack: 19900 current_bet: 100 side_pot_rank: -1 hand: 8c, Qd, | Side_pot1: 0

What action do you want to take as player 0?2

preflop - 1 acts
Board:
Last Action: player_0: 2
Player_0:stack: 19800 current_bet: 200 side_pot_rank: -1 hand: 5s, 4h, | Side_pot0: 0 200 | Main_pot: 0
Player_1:stack: 19900 current_bet: 100 side_pot_rank: -1 hand: 8c, Qd, | Side_pot1: 0

preflop - 0 acts
Board:
Last Action: player_1: 2
Player_0:stack: 19800 current_bet: 200 side_pot_rank: -1 hand: 5s, 4h, | Side_pot0: 0 20000 | Main_pot: 0
+Player_1:stack: 0 current_bet: 20000 side_pot_rank: -1 hand: 8c, Qd, | Side_pot1: 0

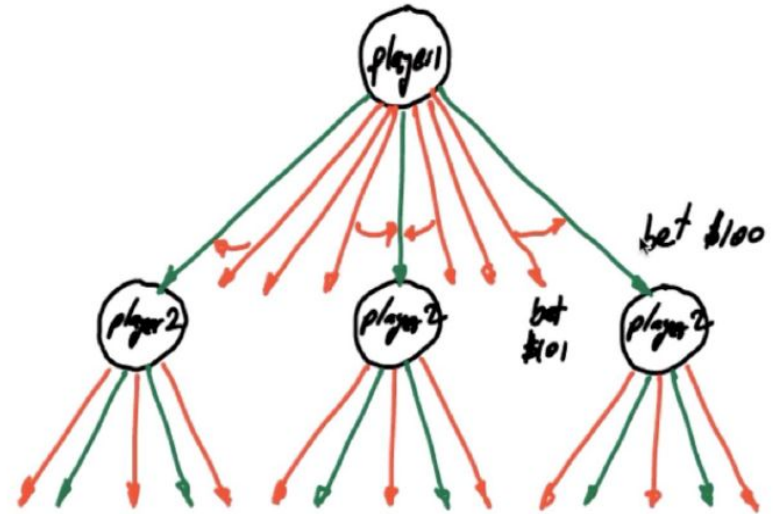
What action do you want to take as player 0?1

river - 0 acts
Board: Jd, Kh, Kd, 8s, Jh,
Last Action: player_0: 1
+Player_0:stack: 0 current_bet: 0 side_pot_rank: -1 hand: 5s, 4h, | Side_pot0: 0 20000 | Main_pot: 0
+Player_1:stack: 40000 current_bet: 0 side_pot_rank: -1 hand: 8c, Qd, | Side_pot1: 0

Current Winnings per player: [-20000.0, 20000.0]
Press Enter to go to the next round.
```

Conclusion and Future Work

- Adaptive Traversing
- Action Abstraction (Discretized Bets)
- Tree Pruning
- Card Abstraction



Source: Iddo Drori, Deep Learning Course,
Summer 2019

Questions?