# COUNTERFACTUAL REGRET MINIMIZATION

By Harris Bubalo

# Motivation

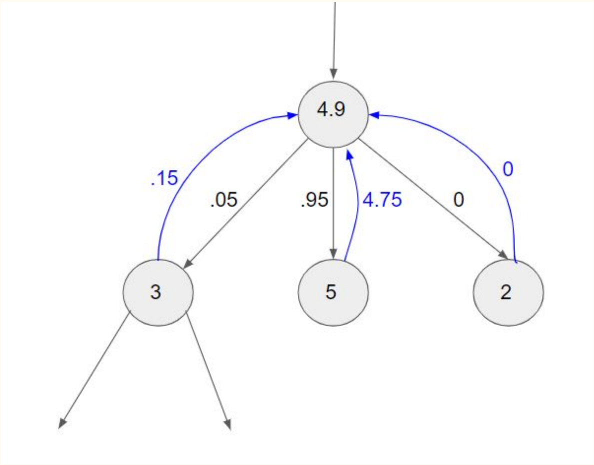
- Games like chess and go are games with **perfect information**
- Meanwhile, card games like poker are **partially observable systems**
- Traditional methods don't work
- Can we make some **$$$?**

# Terminology

- *Relating to or expressing what has not happened or is not the case*
- **Infoset** - “node” in game tree
- **Strategy** - probability distribution across actions
- **Utility** - the payoff for the action
- **Regret** - u(Alternative choice) - u(Current choice)
- Main idea of CFR is to update our strategy at each node according to accumulated regret values

# The CFR Algorithm

1. Run the agent against itself for many iterations (self play algorithm)
2. At each infoset, calculate the strategy using current accumulated regrets
3. Calculate the utility of all possible actions recursively and update accumulated regret
4. Average strategy across all iterations at each node converges to a Nash equilibrium



# POCKET POKER!

https://cfr-poker-web.vercel.app/

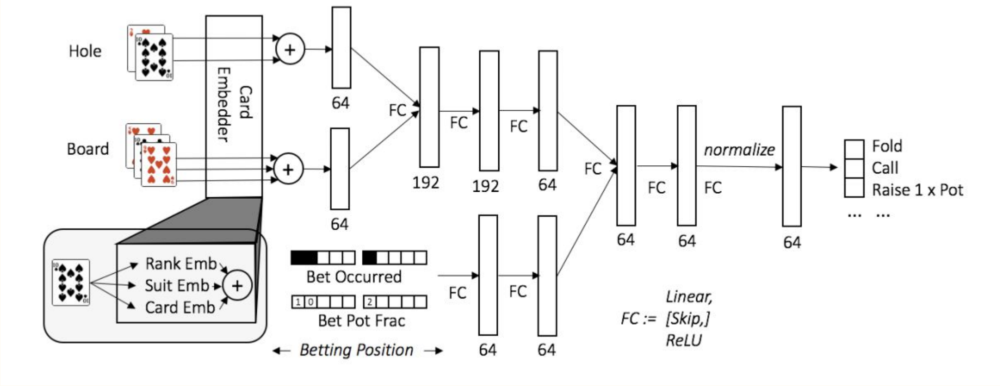
- 20 card deck
- 2 cards to each player
- 1 public card visible to both
- 1 round of betting (fold, check, call, raise)
- Showdown
    - Three-of-a-kind
    - Pair
    - High card

# Results

- **Automated evaluation** - Ran trained player against other player "archetypes"
  - Always large net gain
- **Manual evaluation -** Played against it on my web app
  - Ends up with many more chips than me after a good amount of rounds

# Deep CFR

- **Input:** infoset
- **Output:** predicts regrets
- Replaces the tabular nature of traditional CFR
- My next approach



# THANK YOU!

**Algorithm 4** External Sampling with Stochastically-Weighted Averaging

**Initialize**: $\forall I \in \mathcal{I}, \forall a \in A(I) : r_I[a] \leftarrow s_I[a] \leftarrow 0$
ExternalSampling($h$, $i$):
  **if** $h \in Z$ **then return** $u_i(h)$
  **if** $P(h) = c$ **then** sample $a'$ and **return** ExternalSampling($ha'$, $i$)
  Let $I$ be the information set containing $h$
  $\sigma(I) \leftarrow$ RegretMatching($r_I$)
  **if** $P(I) = i$ **then**
    Let $u$ be an array indexed by actions and $u_\sigma \leftarrow 0$
    **for** $a \in A(I)$ **do**
      $u[a] \leftarrow$ ExternalSampling($ha$, $i$)
      $u_\sigma \leftarrow u_\sigma + \sigma(I, a) \cdot u[a]$
    **for** $a \in A(I)$ **do**
      By Equation $\boxed{4.20}$, compute $\tilde{r}(I, a) \leftarrow u[a] - u_\sigma$
      $r_I[a] \leftarrow r_I[a] + \tilde{r}(I, a)$
    **return** $u_\sigma$
  **else**
    Sample action $a'$ from $\sigma(I)$
    $u \leftarrow$ ExternalSampling($ha'$, $i$)
    **for** $a \in A(I)$ **do**
      $s_I[a] \leftarrow s_I[a] + \sigma(I, a)$
    **return** $u$