**Project Proposal: Model-Free Reinforcement Learning for Blackjack**

**Project Group 4: Chelsea Liu, Michelle Peng, Sherry Xu, Wen Yao**

Blackjack is a widely studied casino card game that presents an ideal testbed for reinforcement learning (RL). The goal of this project is to develop an RL agent that can learn an optimal playing strategy for maximizing long-term rewards in Blackjack. By comparing RL-based strategies with traditional heuristics (e.g., basic strategy), we aim to evaluate the effectiveness of RL techniques in learning optimal decision-making in stochastic environments.

**1. Game Description**

There are two roles, players and dealer, competing in the Blackjack game where. The goal is to have a hand value as close to 21 as possible without exceeding it.

The original cards of each role and the actions they can take are shown in the table:

|  | Player | Dealer |
| --- | --- | --- |
| Original cards | Two face-up card | One face-up card + one face-down card |
| Probable actions | Take turns deciding whether to:   * Hit (draw another card) * Stand (keep the current hand) * Double Down (double the bet and take exactly one more card) * Surrender (forfeit half the bet and end the round early)   (Double Down and Surrender can only be done right after taking the original two cards) | Must:   * hit on 16 or less * stand on 17 or more |

Number cards (2-10) are worth their face value. Face cards (J, Q, K) are worth 10 points. And Aces (A) are worth 1 or 11 points, whichever is more favorable.

Blackjack (Ace + 10-point card) automatically wins 1.5x the bet unless the dealer also has Blackjack (then it’s a tie) (Actually there is no room for strategy optimization in this situation). If there are no Blackjacks, players take turns to take actions. If any player exceeds 21, he or she loses immediately (bust). If the dealer busts, the player wins the round. If both the player and dealer stay under 21, the higher total wins.

**2. Research Objectives**

Traditional Blackjack strategies rely on predefined rules or Monte Carlo simulations. However, a model-free RL approach allows the agent to learn optimal strategies purely through interaction with the environment. So, the objectives of our group is to:

1. Implement a model-free RL Blackjack agent.
2. Implement several models using different RL approaches, including Monte Carlo Learning and Q-Learning.
3. Evaluate RL agents’ performance against standard Blackjack strategies, such as Basic Strategy and Card Counting.
4. Extend the project to multi-agent settings by including dealer behavior and opponent strategies.

**3. Environment Description**

The Blackjack environment follows the standard OpenAI Gym Blackjack-v1 framework, where an agent plays against a dealer in a simplified version of the casino game. The environment consists of the following:

* State Space: The state is represented as a tuple (player’s sum, dealer’s visible card, usable ace flag).
* Action Space: The agent can take two actions: Hit, Stand.
* Reward Function: +1 if the player wins, -1 if the dealer wins, 0 if the round ends in a draw(push).

This structured environment allows the agent to learn through self-play and improve its policy over time. The agent will interact with this environment repeatedly to learn an optimal policy for maximizing rewards.

If time permitted, we will also try some more complex frameworks. The current idea is to set more players or expand the action space as follows:

* New Action Space: The agent can take four actions: Hit, Stand, Double Down, Surrender.
* New Reward Function:
  + +2 (if the player wins with a double down),
  + +1 (if the player wins without a double down),
  + 0 (if the player wins with a double down) ,
  + -0.5 (if the player forfeit),
  + -1 (if the player loses without a double down),
  + -2 (if the player loses with a double down)

**4. Evaluation Metrics**

The trained RL agent will be evaluated based on:

* Win rate compared to the random policy, rule-based policy and dealer’s policy.
* Convergence of Q-values during training.
* Cumulative rewards over multiple test episodes.

Baseline comparisons will include a random policy, a basic strategy player, and the dealer’s fixed policy (hitting until 17).

**5. Expected Outcomes**

* An RL-based agent capable of playing Blackjack optimally.
* Insights into the effectiveness of different RL algorithms in a stochastic card game.
* A comparison between tabular and deep reinforcement learning methods.

Proposal 2: Multi-Agent Reinforcement Learning for Texas Hold’em Poker

Multi-Agent Reinforcement Learning for Texas Hold’em Poker Strategy Optimization

Texas Hold’em Poker is a multi-agent, imperfect information game, making it a complex but exciting challenge for RL. Unlike deterministic board games like chess or Go, Poker requires players to learn from incomplete information, predict opponent behavior, and optimize betting strategies. The goal of this project is to train an RL agent that can compete effectively in simplified Poker scenarios, such as Heads-Up No-Limit (1v1 Poker).

1. Game Description

The goal in Texas Hold’em Poker is to win chips by having the best hand or making opponents fold.

| Betting round | Changes in Cards | Cards |
| --- | --- | --- |
| Pre-Flop | Get two private cards (“hole cards”) and three unrevealed community cards. | 2 hole cards + 3 unrevealed community cards |
| Flop | Reveal the original three community cards | 2 hole cards + 3 community cards |
| Turn | Get the fourth community card | 2 hole cards + 4 community cards |
| River | Get the fifth community card | 2 hole cards + 5 community cards |

In each round, players can:

Check (pass without betting)

Bet (place a wager)

Call (match a previous bet)

Raise (increase the bet amount)

Fold (give up and lose the round)

In the end, if two or more players remain after the final bet, they reveal their hands. The player with the strongest hand (based on Poker hand rankings) wins the pot.

If all other players fold, the last remaining player wins.

2. Research Objectives

•Develop a reinforcement learning agent that learns optimal betting strategies in Texas Hold’em Poker.

•Explore Counterfactual Regret Minimization (CFR), Deep Q-Networks (DQN), and Policy Gradient (PPO/A2C).

•Simulate different player styles (tight/aggressive/passive) and evaluate RL agent adaptability.

•Extend to multi-agent RL by simulating multi-player Poker dynamics.

3. Methodology

1.Game Environment:

•Use OpenSpiel’s Texas Hold’em environment or implement a custom one using LCard.

•Define state representation: hand strength, betting history, opponent actions.

•Define action space: fold, check, call, raise, all-in.

2.Reinforcement Learning Algorithms:

•Deep Q-Learning (DQN): Train an agent to estimate expected values for different bets.

•Counterfactual Regret Minimization (CFR): Optimize agent decisions in imperfect information scenarios.

•Policy Gradient (PPO/A2C): Learn directly from game rewards to improve long-term performance.

3.Evaluation Metrics:

•Win rate against different opponent styles (e.g., random, tight, aggressive).

•Expected Value (EV) per hand as a measure of long-term profitability.

•Adaptability against new opponents (generalization ability).

4. Expected Challenges & Solutions

•Imperfect Information Learning: Implement belief-based state representations to infer opponent behavior.

•Multi-Agent Interactions: Train RL agent using self-play to improve adaptability.

•Exploitability vs. Generalization: Use mixed strategies and Nash Equilibrium approximations to prevent agent exploitation.

5. Expected Outcomes

•A trained Poker AI capable of adapting to different opponent strategies.

•Comparative analysis of CFR vs. DQN vs. PPO in Poker decision-making.

•Insights into how RL can approximate Nash Equilibria in imperfect information games.