Extended Abstract

Motivation The Liar Bar game, popular on Steam since 2024, offers a uniquely demanding environment for imperfect-information reinforcement learning: at every turn, players must decide how many cards to play—and whether to bluff—without knowing opponents' hands, then pay a single "Russian Roulette" penalty if their deception is exposed. This combination of hidden information, sequential bluffing, and high-risk, one-shot punishment creates a rich tapestry of strategic and psychological pressures that mirror real-world decision dilemmas. As a result, Liar Bar serves as an ideal crucible for testing how well algorithms like CFR and NFSP can learn to balance risk versus reward, implicitly model other agents' beliefs, and converge toward equilibrium behavior under extreme uncertainty.

Method We implement two different approaches: Counterfactual Regret Minimization(CFR) to approximate Nash equilibrium via regret-matching over compact information sets, and Neural Fictitious Self-Play(NFSP), which interleaves Double-DQN value updates with supervised learning of average policies. We also design a per-round reward which effectively improves the agents' performance. For evaluation, we use exploitability and empirical performance (head-to-head matches win-rate) to comprehensively measure the agents' performances. We also use two baseline models (random and heuristic) to compare with CFR and NFSP methods.

Implementation We implement a custom *Liar Bar* environment faithful to the official rules and game logic. Two variants are trained via self-play: (1) a 2-player mini-game with roulette disabled, and (2) the full 4-player mode with roulette enabled. The discrete action space (10 moves) is uniformly encoded as integers 0–9. For CFR, we use external-sampling traversals over a 10-action discrete space, training on 2/3/4 player games and logging average regret. For NFSP, we implement the dual-network architecture with a 34-dimensional normalized state vector input in 4-player mode. NFSP models has access to full history of actions and roulette status. NFSP agents employ two-layer MLPs (64 units each), dual replay buffers (100 k RL, 400 k SL), and gradient clipping. The ϵ-greedy fraction anneals from 1.0 to 0.1 over 400 k episodes ($\approx$20 h on a single GPU). Both approaches are trained with four independent seeds per configuration, and until convergence of their strategy profiles, enabling direct comparison of their convergence rates, exploitability, and head-to-head performance.

Results Across both two- and four-player Liar Bar settings, our results show that while tabular CFR and NFSP both significantly outperform random and heuristic baselines, their relative strengths diverge with increasing game complexity: in the two-player, roulette-off setting, CFR slightly outperforms NFSP (55.1% vs. 44.9%), benefiting from its ability to compute near-exact equilibrium strategies. However, in the four-player, roulette-on setting, NFSP's history-aware neural strategy consistently dominates, achieving win rates exceeding 60% against baselines, while CFR underperform comparing to NFSP (32% vs. 18%). Exploitability analysis further reveals that CFR has a regret of 17.5%, compared to NFSP's nearly unexploitable regret of 0.7%. Ablation studies demonstrate that using Double DQN stabilizes learning by reducing overestimated Q-values and inter-run variance, while a per-round reward signal improves convergence speed and boosts final win rates by approximately 15%.

Discussion Our study also has several limitations. We were unable to conduct human–AI matches—particularly against expert players—due to time constraints, leaving open the question of how well our agents' bluffing strategies align with human psychology and adapt to nuanced human behavior. Additionally, we are limited to tabular CFR; future work should incorporate stronger regret-minimization methods such as Deep CFR or Monte Carlo CFR to fully characterize the trade-offs between tabular and deep approaches in environments like Liar Bar.

Conclusion We present the first reinforcement-learning agents for Liar Bar, showing that tabular CFR excels in small, near-zero-sum games but cannot accommodate richer multi-agent dynamics. In contrast, NFSP's deep self-play framework not only outperforms CFR in complex scenarios but also remains nearly unexploitable. Crucially, the integration of Double-DQN updates and per-round reward design accelerates and stabilizes learning. Our work thus lays a foundation for future AI research in multi-player bluffing games and other high-uncertainty, imperfect-information domains.

Bluff and Learn: Comparing CFR and NFSP in Liar Bar

Cici hou
Department of Computer Science
Stanford University
xhou@stanford.edu

Louise Li
Department of Mathematics
Stanford University
louisely@stanford.edu

Phillip Miao
Department of Computer Science
Stanford University
pmiao@stanford.edu

Abstract

We present the first systematic AI framework for playing Liar Bar, a challenging multi-player bluffing game with imperfect information, sequential deception, and stochastic penalties. By implementing and comparing Counterfactual Regret Minimization (CFR) and Neural Fictitious Self-Play (NFSP) across both two- and four-player game modes, we highlight the critical trade-offs between tabular and neural approaches to equilibrium learning. Our results show that while CFR performs well in simplified settings, NFSP—enhanced with Double DQN and dense per-round rewards—achieves superior generalization, faster convergence, and near-zero exploitability in complex scenarios. This work not only introduces the first AI agents for Liar Bar but also provides a robust benchmark for future research on scalable, strategy-aware learning in high-risk, imperfect-information environments.

1 Introduction

The Liar Bar game presents a compelling environment for exploring reinforcement learning (RL) strategies, particularly due to its elements of deception, partial observability, and high-stakes decision-making. Released on Steam in October 2024, the game has surged in popularity—surpassing titles like "Diablo IV" in user engagement and accumulating over 42,000 player reviews with a 91% approval rating—making it a prominent testbed for bluffing and strategy in multiplayer settings (MeriStation, 2024). In this project, we aim to train RL agents using two prominent algorithms—Counterfactual Regret Minimization (CFR) and Neural Fictitious Self-Play (NFSP)—to navigate the complexities of Liar Bar and compare their performance.

Liar Bar is a multiplayer card game designed for 2 to 4 players, where the objective is to be the last player remaining. Each round begins with the revelation of a table card, determining the "innocent" card type for that round, while all other cards are considered "liars." Players take turns playing 1 to 3 cards face down, claiming them to be of the innocent type, or challenging the previous player's claim by calling "LIAR." If a challenge occurs, the accused player's cards are revealed: if any are liars, the accused faces a Russian Roulette penalty; otherwise, the challenger does. This mechanic introduces a layer of psychological strategy, as players must decide when to bluff and when to call out potential deception.

Given the game's structure, Liar Bar can be modeled as an extensive-form game with imperfect information, making it suitable for analysis using CFR and NFSP. CFR is an iterative algorithm that minimizes regret over time, converging towards a Nash equilibrium in two-player zero-sum games (Zinkevich et al., 2007; Tammelin, 2014). It operates by simulating playthroughs of the game tree, updating strategies based on counterfactual regrets—essentially, the difference between the actual outcome and what could have occurred had a different action been taken. This method has been successfully applied to games like poker, where hidden information and bluffing are central elements.

NFSP, on the other hand, combines reinforcement learning with fictitious self-play to approximate Nash equilibria in large-scale games (Heinrich and Silver, 2016). It employs a dual-network approach: one network learns the best response strategy through reinforcement learning, while the other averages these strategies over time using supervised learning. This allows NFSP to handle the complexities of imperfect-information games without requiring handcrafted abstractions, as demonstrated in applications to games like Leduc Hold'em (Qu et al., 2022).

By implementing both CFR and NFSP in the context of Liar Bar, we aim to evaluate their effectiveness in handling the game's unique challenges, such as deception, risk assessment, and partial observability. Comparing these methods will provide insights into their respective strengths and limitations, contributing to the broader understanding of RL strategies in complex, imperfect-information environments.

2 Related Works

2.1 CFR in Imperfect-Information Games

Counterfactual Regret Minimization (CFR), introduced by Zinkevich et al. Zinkevich et al. (2007), is a foundational algorithm for computing Nash equilibria in two-player zero-sum extensive-form games with imperfect information. By iteratively minimizing *counterfactual regret*, it provably converges toward equilibrium strategies. CFR+ Tammelin (2014) and variants such as discounted CFR improve convergence speed, enabling solutions for games like Heads-Up Limit Hold'em Bowling et al. (2015). Landmark poker systems such as DeepStack Moravčík et al. (2017) and Libratus Brown and Sandholm (2018) leverage CFR in conjunction with search and deep function approximation to surpass human performance in no-limit Hold'em. CFR variants have also been explored in multi-player bluffing games like Liar's Dice He et al. (2022), although scaling beyond two-player zero-sum settings requires further abstraction and algorithmic innovation.

2.2 NFSP and Fictitious Self-Play

Neural Fictitious Self-Play (NFSP) Heinrich and Silver (2016) combines reinforcement learning and fictitious play, maintaining a best-response policy via deep RL alongside an averaged policy through supervised learning. NFSP has been empirically validated on games like Leduc and Limit Texas Hold'em, achieving exploitability near CFR baselines without explicit abstraction. Heinrich et al. Heinrich et al. (2015) detail the theoretical underpinnings of FSP in extensive-form games, which NFSP builds upon. Recent improvements in Liar's Dice using local-regret FSP demonstrate continued advancements in neural regret-minimization techniques He et al. (2022).

2.3 Applications to Liar's Dice and Guandan

Imperfect-information bluffing games such as Liar's Dice and Guandan represent more complex environments than two-player poker. While CFR has been used in simplified Liar's Dice variants, NFSP and newer FSP methods yield lower exploitability and better scalability He et al. (2022). Multi-player games like Guandan—characterized by cooperation, deception, and large state spaces—are ill-suited for pure CFR. Instead, systems like DanZero Lu et al. (2023) and GuanZero Yanggong et al. (2024) combine Monte Carlo self-play and behavior regularization to approach or exceed human-level performance, demonstrating that hybrid deep RL methods can handle the intricacies of multi-agent, imperfect-information games effectively.

3 Methods

We start with a mini-game of 2-player and roulette off version of Liar Bar to build vanilla CFR and NFSP models to test the plausibility of our methods, then we elaborate to the 4-player and roulette on version.

3.1 State and Action Encoding Designs

To support both CFR and NFSP training in the Liar Bar game environment, we design compact and informative encodings for states and actions. The action space consists of ten legal moves, encompassing all valid combinations of real and fake cards played in a single action, as well as the special action of calling "LIAR." These include: playing 1 to 3 fake cards, 1 to 3 real cards, combinations of real and fake cards (e.g., 1 fake + 1 real), and the liar call. We encode these actions as discrete integers ranging from 0 to 9, allowing for efficient indexing and compatibility with both tabular and neural representations.

For the state representation, we differentiate between encodings used for CFR and NFSP due to their distinct architectures. CFR relies on tabular representations, which necessitate a compact state space for tractability, while NFSP utilizes deep neural networks, making it feasible to encode richer and more temporally extended information.

In the CFR setting, the state includes only essential information: the player's current hand (expressed as counts of real and fake cards), the number of cards held by each opponent, and the number of cards played in the most recent move. This minimal state enables the agent to infer whether a "LIAR" call is warranted, without incurring the combinatorial explosion that would result from tracking full action histories. For example, the state can be represented as a tuple of the form:

```
(real, fake, oppo 1 card, [oppo 2, oppo 3 if applicable], last move)
```

In contrast, the NFSP encoding incorporates a more comprehensive view of the game, as shown in Figure 1. For the two-player version (roulette off), the state is represented as a 17-dimensional normalized vector. This includes the player turn order via one-hot encoding, normalized counts of real and fake cards in the player's hand, the opponent's total card count, and both players' recent histories over the last four rounds. The player's history captures claimed real and fake cards per

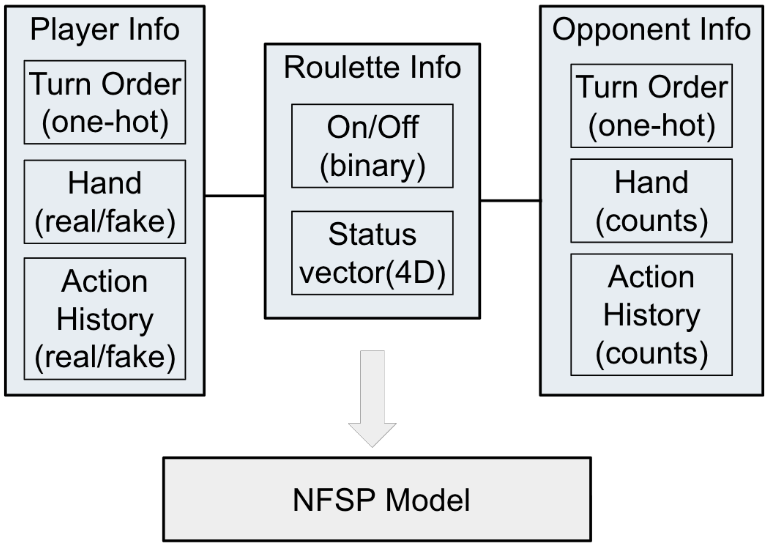


Figure 1: State Encoding Design for 4-player version NFSP model

round, while the opponent's actions are encoded as cards played in each round, all normalized for input into the neural network.

The four-player NFSP version further extends this representation to a 34-dimensional vector. Player 0 is always the agent itself, and subsequent players (Player 1 to Player 3) are ordered clockwise. The encoding includes one-hot indicators of which player started the current round, the agent's hand composition, the remaining card counts of all opponents (set to zero if a player is eliminated), and an indicator for whether roulette is enabled. Additionally, we include roulette-specific status information, such as the probability of elimination based on shots taken. The final portion of the state includes normalized round-by-round histories of claimed cards for the agent and normalized opponent card plays for each of the other players.

This encoding scheme balances the trade-off between expressiveness and computational efficiency. It allows CFR to operate over a compact, abstracted space while enabling NFSP to leverage deeper strategic patterns through richer input features and recurrent history. Together, these encodings ensure fair and consistent comparison between the two methods across experimental settings.

3.2 Counterfactual Regret Minimization (CFR)

We approximate a Nash equilibrium for Liar Bar by running external-sampling counterfactual-regret minimisation (ES-CFR) on the full information-set tree. At a high level, each depth-first traversal of the game tree performs three tasks:

1. **Enumerate and evaluate the acting player's legal moves.** For the current information set I we enumerate the encoded actions $\mathcal{A}(I) \subset \{0, \ldots, 9\}$. An index encodes whether the move is Call Liar or Play k cards with (real,fake) = (r,f).
2. **Compute counterfactual utilities and update regrets.**
   Let
   $$\pi_{\text{opp}} \;=\; \prod_{\substack{\text{chance events}\\ \text{and opponents' moves}}} \Pr(\text{event})$$
   be the probability that chance and all opponents (but not the target player) reach I. Because π_{opp} is the same for every $a \in \mathcal{A}(I)$, we factor it out and scale the instantaneous regret
   $$\Delta r_a \;=\; \pi_{\text{opp}}\Big(u_a \;-\; \sum_{b \in \mathcal{A}(I)} \sigma_b \, u_b\Big),$$
   where u_a is the counterfactual value of choosing a and σ is the current strategy produced by regret-matching on the positive part of the cumulative regret vector.
3. **Sample a single action at opponents' nodes.** When the recursion reaches an opponent, we draw one action from their mixed strategy σ instead of enumerating all moves, keeping each iteration linear in $|\mathcal{A}(I)|$.

After each outer iteration we reset the root position with a new random seed, so the traversal covers a representative set of deal permutations.

3.3 Neural Fictitious Self-Play (NFSP)

Our NFSP implementation follows Heinrich and Silver's two-network architecture, adapted to the hidden-information dynamics of Liar Bar (**?**). Each agent maintains (i) a Q-network that continually learns a near-best-response to opponents' current average strategies, and (ii) an SL-network that tracks the agent's own historical behaviour so the overall policy drifts toward a smeared-out Nash equilibrium.

NFSP specifics At every decision point the agent flips a biased coin: with probability η it follows an ϵ-greedy best response from the Q-network, and with probability $1 - \eta$ it samples directly from the SL-network's softmax policy. The training diagram is shown in Figure 2. Trajectories generated under this mixed strategy feed two replay buffers that serve complementary learning objectives.

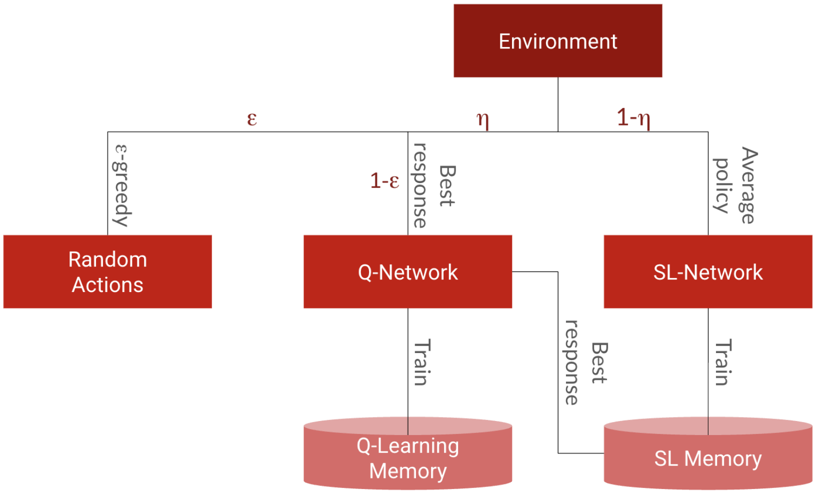


Figure 2: NFSP decision flow: with probability η the agent follows an ϵ-greedy best response from the Q-network, otherwise it samples its SL-network average policy; resulting transitions populate separate RL and SL memories for Double-DQN and behaviour-cloning updates.

1. **Off-policy RL buffer** (D_{RL}). Every transition (s, a, r, s') is stored, irrespective of how the action was selected, because all experience is useful for temporal-difference updates. After each environment step a minibatch from D_{RL} is used to perform a Double-DQN update on the online Q-network with loss

$$\mathcal{L}_{\text{RL}}(\theta^{Q_{\text{on}}}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}_{\text{RL}}}\big[r + \gamma Q_{\text{tgt}}\big(s', \arg\max_{a'} Q_{\text{on}}(s', a')\big) - Q_{\text{on}}(s, a)\big]^2.$$

   while the target network performs soft update through Polyak averaging.

2. **Reservoir SL buffer** (D_{SL}). Only state–action pairs produced by the ϵ-greedy branch are appended, so the buffer represents the agent's historical best responses. After each environment step, a batch of random sample from D_{SL} is drawn to minimize a cross-entropy loss on the SL-network, distilling the running average strategy. The cross-entropy loss is

$$\mathcal{L}_{\text{SL}}(\theta^{\pi}) = -\,\mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{SL}}}\ \log \pi(a \mid s; \theta^{\pi}).$$

This dual-buffer design allows NFSP to interleave off-policy value learning with on-policy behaviour cloning, gradually steering the mixture policy toward an approximate Nash equilibrium.

Double DQN & Gradient Clipping In early experiments (see Section 6.3), our Q-learning updates exhibited instability: severely overestimated Q-values in the initial training stages slows overall convergence. To address this, we employ the Double DQN update—which decouples action selection and evaluation to reduce overestimation bias—and apply gradient clipping to cap the size of each Q-network update, thereby enhancing training stability and speeding convergence.

Reward Design We employ sparse but scaled payoffs to guide both win maximization and risk management. In two-player games, victories yield $+1$ and losses 0. In four-player games, we award $+1.2$ for a win and apply a -0.2 penalty for each roulette pull (so that six pulls exactly offset one victory). As our ablation study demonstrates (see Section 6.3), this modest -0.2 penalty substantially speeds up convergence by providing denser feedback and stabilizing training.

4 Evaluation

To assess the performance and robustness of our trained RL agents, we evaluate along two complementary dimensions: exploitability and empirical performance. In empirical performance, we compare with two baseline models to thoroughly evaluate our models.

4.1 Exploitability

Exploitability measures a policy's theoretical vulnerability to an optimal adversary (Timbers et al., 2022). Formally, let

$$u(\mu, \nu) \;=\; \mathbb{E}\big[\text{total payoff when using policy } \mu \text{ against } \nu\big].$$

The *best-response* to policy π is

$$\mathrm{BR}(\pi) \;=\; \arg\max_{\mu} \; u(\mu, \pi),$$

and the exploitability of π is defined as

$$\varepsilon(\pi) \;=\; u\big(\mathrm{BR}(\pi),\, \pi\big) \;-\; u\big(\pi,\, \pi\big).$$

Intuitively, $\varepsilon(\pi)$ quantifies how much an optimal opponent can gain by exploiting the weaknesses of π. A low exploitability indicates that the policy is nearly unexploitable and hence close to a Nash equilibrium.

4.2 Empirical Performance

We have two baseline models to compare with:

- **Random model.** At each decision point, this agent selects uniformly at random from the set of all legal actions. This establishes a performance "floor" that any learned policy should surpass.
- **Heuristic model.** Similar to the random model, but it excludes obviously irrational moves—specifically, actions that simultaneously give both real and fake cards (the "LIAR" move), which, while legal, wastes real cards. This baseline incorporates minimal domain knowledge and sets a higher bar than pure randomness.

We measure empirical performance by having each policy π play N head-to-head matches against given opponents (random, heuristic, or another RL agent) and computing

$$\mathrm{WinRate}(\pi, \mathrm{opp}) \;=\; \frac{\#\{\text{wins by } \pi\}}{N}\,. \tag{1}$$

This captures average-case effectiveness in direct competition.

Exploitability quantifies worst-case vulnerability: how severely an optimal adversary can defeat the policy; while wWinrate captures practical, averagecase strength against typical opponents. By combining both metrics, we are able to test not only if our agents are strong in practice (high win-rate) but also if they are resilient against strategic exploitation (low exploitability).

5 Experimental Setup

5.1 Training

CFR Counterfactual Regret Minimization struggles to scale as game depth and branching increase, so we deliberately omitted the roulette mechanic during training to keep the abstraction tractable. Since roulette outcomes depend purely on chance and do not fundamentally alter bluff–call dynamics, this simplification is assumed to have minimal impact on overall strategy. For the two-player mini-game, we trained CFR on the full 2-player game tree without roulette. To support four-player roulette-on play, we precomputed CFR trees for 2-, 3-, and 4-player single-round games (also without roulette). At runtime, whenever a roulette penalty eliminates a player, the agent seamlessly switches to the next lower-player CFR model until only one remains.

For each of these CFR trees, we ran increasing iteration budgets and monitored average counterfactual regret, selecting the smallest count that still showed clear convergence: 10,000 iterations for two players ($\approx$ 10 min), 50,000 for three ($\approx$ 1.5 h), and 200,000 for four ($\approx$ 5 h). Each outer iteration began from a freshly shuffled deal, and within each depth-first traversal we enumerated all actions for the acting player while sampling a single action at opponents' nodes. Regrets and reach-weighted average strategies were stored in a compact information-set table.

NFSP All NFSP agents use two-layer MLPs (64 units per hidden layer, ReLU activations) for both the Q-network and SL-network. We train for 400k episodes ($\approx$ 4 h), storing transitions in an off-policy replay buffer of capacity 100k (RL) and a reservoir buffer of capacity 400k (SL). Discounting is undiscounted ($\gamma = 1$) due to short horizon, and target-network weights are softly updated each step with $\tau = 0.005$. The anticipatory parameter is set to η = 0.1, and the ϵ value for the ϵ-greedy best-response linearly anneals from 1.0 to 0.1. We use batch sizes of 64 for RL updates and 256 for SL updates, with both optimizers as Adam at a learning rate of 1e-3.

5.2 Evaluation

Exploitability To evaluate exploitability, We choose NFSP as the exploitative agent because of its ability to approximate best responses through reinforcement learning, while still maintaining stability via supervised learning. This dual-process architecture enables NFSP to effectively adapt to and exploit the fixed behavior of opponents, making it a practical choice for estimating best-response performance in empirical settings. Both for CFR and NFSP, we freeze three trained models and train a new NFSP agent to exploit them. Then, we record the win-rate of the new NSFP agent competing with the three frozen models as best response. We also conduct self-play experiments for both models. Each model is trained and evaluated over 10 independent runs, and for each run, we perform 1,000 evaluation games. During each evaluation, player seats are randomly shuffled to mitigate the confounding effects of seating position.

Empirical Performance We perform head-to-head evaluations in two modes. In the two-player and roulette-off variant, each pairing is tested over 10 000 games, with turn order shuffled each game. For the full four-player liar-bar with roulette, each configuration is played for 1 000 games. We compare the standard stochastic CFR, its deterministic greedy variant O-CFR (which always selects the highest-probability action), NFSP, and two baseline agents (random and heuristic) comprehensively. In all experiments, turn order is randomized at each episode. For *1 vs. 3* scenarios seating order is irrelevant; for *2 vs. 2* scenarios we record both side-by-side (S) and opposite (O) seatings to assess robustness against positional effects.

6 Results and Analysis

6.1 Self-Play and Head-to-Head Performance

Table 1 summarises the average win-rates obtained from 2- or 4-player games across different match-up settings, with the turn order randomly shuffled.

Two-player, roulette-off Both CFR and NFSP converge to strong bluff–call strategies, comfortably beating the heuristic and random baselines. CFR even edges out NFSP in the direct face-off (55.1% vs. 44.9%), highlighting tabular CFR's ability to compute near-exact equilibrium probabilities when the state space is small and the game is close to zero-sum.

Four-player, roulette-on The picture changes in the four-player roulette-on game. Here the strategic surface is far richer—players must juggle multi-way bluffing, dynamic risk from the roulette penalty, and a larger action history. NFSP's neural agents consistently dominate: a single NFSP paired with three random or heuristic opponents wins over 60% of the time, and two cooperating NFSPs still out-perform teams of CFR variants. The tabular CFR (and its optimal-strategy variant O-CFR) struggle to generalize because their information sets omit the detailed temporal patterns that NFSP's network can exploit. As a result, CFR's win-rates collapse once more players and roulette dynamics are introduced, whereas NFSP remains robust even when numerically out-matched.

It is worth noting that the CFR that always selects the best strategy (O-CFR) enjoys a markedly higher raw win-rate than the probabilistically random CFR (CFR), because it always plays the single highest-value action. However, this deterministic behavior also makes it extremely predictable. As a consequence, O-CFR is far more susceptible to targeted exploitation once opponents learn its fixed pattern. These contrasting behaviors motivate the deeper analysis in the next subsection, where we quantify exactly how exploitable each algorithm is.

	Match-up Settings			Average Win Rate	Average Win Rate
Game settings	**Model Type A**		**Model type B**	**of Model A**	**of Model B**
2 Players No Roulette	**1 Random**	**V.S.**	**1 Heuristic**	41.0%	**59.0%**
	1 CFR		**1 Random**	**77.1%**	22.9%
			1 Heuristic	**67.1%**	32.9%
	1 NFSP		**1 Random**	**73.3%**	26.7%
			1 Heuristic	**62.1%**	37.9%
	1 CFR		**1 NFSP**	**55.1%**	44.9%
4 Players Use Roulette	**2 Random(S)**		**2 Heuristic(S)**	19.9%	**30.1%**
	2 Random(O)		**2 Heuristic(O)**	17.8%	**32.2%**
	1 CFR		**3 Random**	**56.6%**	14.5%
			3 Heuristic	**37.8%**	20.7%
			3 NFSP	9.9%	**30%**
	1 O-CFR		**3 Random**	**66.1%**	8.5%
			3 Heuristic	**49.6%**	12.6%
			3 NFSP	16.7%	**27.8%**
	1 NFSP		**3 Random**	**68.9%**	10.4%
			3 Heuristic	**61.1%**	13.0%
			3 CFR	22.2%	**25.9%**
			3 O-CFR	8.8%	**30.4%**
	2 NFSP		**2 CFR(S)**	**30.5%**	19.5%
			2 CFR(O)	**32.7%**	17.3%
			2 O-CFR(S)	18.7%	**31.3%**
			2 O-CFR(O)	20.0%	**30.0%**

Table 1: Match-up settings and win rate across different game configurations and models.
*S: Same models seated side by side; O: Same models seated opposite each other; O-CFR: CFR with optimal strategy.

6.2 Exploitability Analysis

We estimate exploitability by freezing three trained copies of a policy and training a fresh NFSP agent to act as a best response. The resulting statistics are reported in Table 2.

Models	Self-Play Win Rate Mean	Self-Play Win Rate Std	Best Response Win Rate	Exploitability / Regret
CFR	25%	±1.36%	42.5%	17.5%
NFSP	25%	±1.32%	25.7%	0.7%

Table 2: Exploitability analysis for CFR and NFSP.

In self-play both CFR and NFSP achieve the expected 25% win rate with low variance, confirming that each algorithm has reached a stable equilibrium in four-player settings. However, the best-response probe reveals a stark contrast: the exploiting agent wins 42.5% against CFR but only 25.7% against NFSP. Put differently, CFR carries an exploitability (average counterfactual regret) of **17.5%**, signaling significant strategic weaknesses that a targeted adversary can mine. NFSP's corresponding figure is a mere 0.7%, statistically indistinguishable from the average and therefore essentially unexploitable in practice.

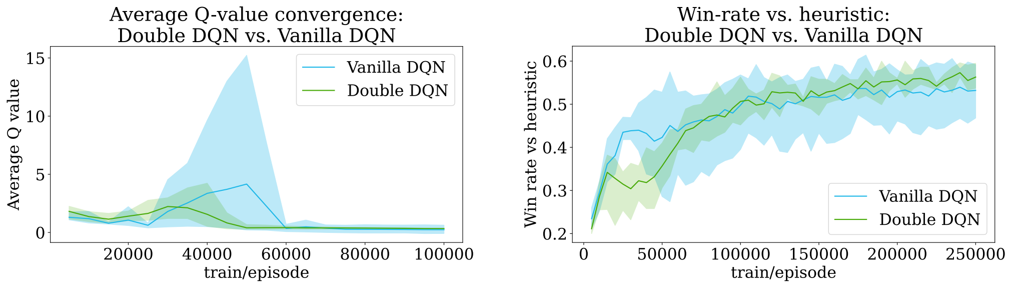
Together, these findings suggest a clear trade-off. CFR is highly effective in small, nearly zero-sum scenarios but scales poorly as strategic complexity grows, leaving it open to exploitation. NFSP, with its history-aware neural architecture, maintains strong empirical performances and exhibits near-zero exploitability across the full four-player Liar Bar setting.

6.3 Ablation Study

We isolate the contribution of (i) the **Double-DQN critic** and (ii) the **per-round reward signal** through targeted ablations.

Double DQN vs. vanilla DQN. Figure 3a plots the average Q-value during training. The vanilla critic overshoots sharply in the first 60 k episodes, peaking at more than 8x the eventual steady-state value. Replacing the max operator with Double DQN almost halves this spike and flattens subsequent oscillations, indicating tighter value estimates and faster stabilisation.

These value-scale effects translate directly into playing strength (Figure 3b). Vanilla DQN enjoys a brief head start, but its performance saturates around 53% win-rate and exhibits wide dispersion (min–max band ≈ 20%). Double DQN continues to climb, converging near 56% and—crucially—compressing the inter-run variance to under 8%. Thus the double estimator not only curbs Q over-estimation but also regularizes learning, preventing the "lucky-run / catastrophic-run" instability seen with the single estimator.



(a) Average Q-value convergence

(b) Win-rate vs. heuristic

Figure 3: Side-by-side comparison of NFSP agents with and without Double DQN: (a) Q-value convergence and (b) win-rate against the heuristic baseline. Shaded regions show min–max across four independent runs.

Reward-shape ablation. Figure 4 compares our dense *per-round* reward to a *sparse, terminal-only* variant. The sparse setup lags throughout training and finishes roughly 15% absolute win-rate lower after 250 k episodes. The denser signal accelerates learning—reaching 0.45 win-rate almost four times as fast—and ultimately yields much higher win-rate. We attribute this gain to the extra temporal credit: NFSP receives informative feedback every round rather than only at game termination, allowing the RL buffer to bootstrap value estimates over shorter horizons.

Takeaway. Double DQN stabilizes training by curbing Q-value overestimation and significantly reducing inter-run variance. Our per-round reward design delivers much faster convergence by providing denser feedback at every step. Together, these enhancements yield NFSP agents that learn more reliably and reach more than 15% higher win-rates in far fewer episodes, proving the effectiveness of our model and reward design.

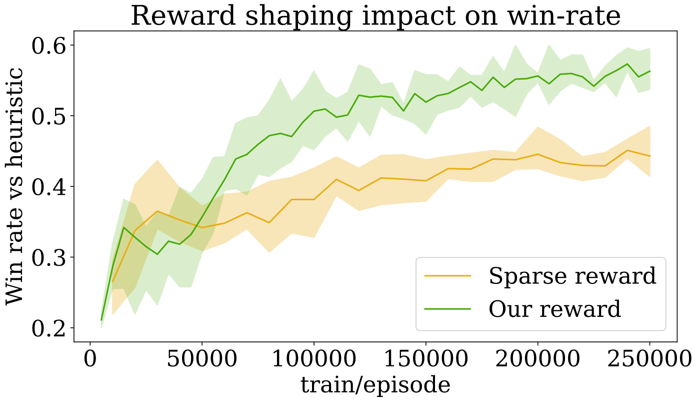


Figure 4: Sparse end-of-game reward ("single reward") versus denser per-round reward. Each curve is the mean win-rate of four independent NFSP agents, sampled every 5 k episodes up to 250 k; shading shows the min–max band. Per-round shaping speeds learning and plateaus roughly 15 percentage-points higher while also reducing variance.

7 Discussion

Limitations While our findings provide strong empirical and theoretical support for the superiority of NFSP in multi-agent bluffing scenarios, several limitations remain. First, we did not conduct any human evaluation due to time constraints. Direct comparisons between trained agents and human players—especially skilled or expert-level ones—would offer valuable insights into how these models generalize to real-world behavior and bluffing psychology. Second, our comparison was limited to tabular CFR. While this provides a solid baseline, future work should include stronger baselines such as Deep CFR and Monte Carlo CFR to more comprehensively evaluate the trade-offs between deep and tabular regret minimization frameworks in complex games like Liar Bar.

Broader Impact To our knowledge, this is the first AI system designed to play Liar Bar, a multiplayer imperfect-information card game with hidden hands, bluff dynamics, and stochastic punishment. Our framework offers a reproducible benchmark for training and evaluating AI agents in this domain. Beyond that, we provide a detailed comparison between two foundational approaches—CFR and NFSP—on a realistic, multi-agent environment. We demonstrate that neural agents, particularly those augmented with Double DQN and shaped rewards, can outperform tabular opponents both in win rate and exploitability. Our ablations illustrate how Double DQN improves training stability and how dense, per-round rewards accelerate convergence significantly. These findings are likely to generalize to other multi-agent games where learning stable policies under partial observability is critical.

Challenges and Lessons Learned Implementing the full Liar Bar environment from scratch proved to be nontrivial. The complexity of managing sequential actions, dynamic player elimination, and partial information demanded meticulous game engine design. Developing a bug-free NFSP agent was also a challenge: maintaining two separate networks, two replay buffers, and interleaved learning loops required careful debugging and testing. On the CFR side, long game horizons and large state spaces made it difficult to apply tabular methods directly, particularly in four-player roulette-on settings. We mitigated this by training separate 2-, 3-, and 4-player CFR models in simplified subgames (e.g., without roulette) and switching them dynamically as the game progressed. Through this process, we learned that investing in clean, modular environment code is essential for reproducibility and that ablation studies are invaluable for uncovering which architectural and reward-design choices drive performance. Overall, the project was a technically intensive yet highly rewarding experience, offering deep insight into the mechanics of imperfect-information learning at scale.

8 Future Research

To further improve model performance and robustness in the *Liar Bar* game, several research avenues remain open. First, we plan to enhance the reward design by incorporating richer and more frequent feedback signals. This denser reward structure could provide stronger learning gradients and improve convergence speed, especially in sparse-reward scenarios.

Second, we aim to explore more scalable variants of CFR, such as Deep CFR and Monte Carlo CFR (MC-CFR). These approaches are better suited for large-scale imperfect information games and may offer improved generalization and computational efficiency in the four-player setting with complex state representations.

Finally, we propose conducting human–AI evaluation. By pitting trained agents against human players, we can assess whether the models demonstrate strategic competence beyond simulated benchmarks and determine their practical competitiveness in real-world gameplay.

9 Conclusion

In this work, we have presented the first systematic comparison of tabular *external-sampling CFR* and deep-RL *NFSP* on the newly released multiplayer bluffing game *Liar Bar*. Our key contributions are:

- **A unified Liar Bar AI framework** supporting 2–4 players with and without roulette, complete with compact state/action encodings, a turn-shuffled evaluation harness, and both CFR and NFSP implementations.
- **Extensive empirical analysis** demonstrating that CFR achieves near-equilibrium performance in two-player (zero-sum) play but struggles to generalize once multi-way bluffing, history, and roulette risk are introduced, whereas NFSP's history-aware neural policies remain strong and near-unexploitable across all configurations.
- **Exploitability probing** that quantifies each method's theoretical vulnerabilities via best-response training, revealing that NFSP yields effectively zero exploitability (<1% regret) even in four-player roulette-on games, while CFR's average regret exceeds 17%, highlighting its brittleness outside small, near-zero-sum domains.

Beyond these algorithmic insights, our ablation studies show that combining Double-DQN value estimation with per-round reward shaping substantially stabilizes and accelerates NFSP learning, boosting asymptotic win-rates by over fifteen percentage points.

Taken together, these findings underscore a trade-off that is likely to generalize beyond *Liar Bar*. Pure regret minimization, when armed with a compact abstraction, remains highly competitive in small, near–zero-sum sub-games. Yet in richer multi-player settings with long action histories, the additional expressiveness of deep function approximation—and the ability to distil that knowledge into an averaged policy—confers a decisive advantage in both strength and robustness. Future work will explore Deep CFR variants to bridge this gap, richer reward shaping to accelerate NFSP, and human-versus-AI evaluations to benchmark real-world playability.

Looking forward, we see promising extensions in hybridizing CFR with function approximation (e.g., Deep CFR) and validating our agents against human players to assess real-world bluffing competence. We hope this work serves as a foundation for future AI research in multi-player bluffing games and other rich imperfect-information domains.

10 Team Contributions

- **Collective planning.** All three authors met regularly to brainstorm the project scope, agree on the comparative focus between CFR and NFSP, and iteratively refine the overall research design, experimental protocol, and paper narrative. High-level decisions—such as the final reward structure, iteration budgets, and evaluation baselines—were reached by consensus.
- **Cici Hou:** Implemented the full Liar Bar game logic (card representations, turn mechanics, roulette design) and wrote the complete CFR training pipeline that can be applied to 2/3/4-player games—including information-set encoding, regret updates, convergence diagnostics, and batch evaluation scripts.

- **Louise Li:** Built the unified `LiarBarEnv` wrapper used by both algorithms, designed the compact state and action encodings, and implemented the evaluation harness (self-play, head-to-head, best-response training, and data logging). She also designed the exploitability evaluation framework.
- **Phillip Miao:** Implemented the NFSP architecture (Q-network, SL-network, dual replay buffers, Double-DQN updates, gradient clipping) and undertook hyper-parameter tuning, training, and ablation studies. He generated the learning-curve figures and integrated NFSP with the evaluation framework.

Work was divided to balance code, experimentation, and writing responsibilities; each member contributed equal effort to debugging, result analysis, and manuscript preparation.

Changes from Proposal We pivoted our project topic from training an RL agent for the card game Guandan to another trending card game Liar Bar.

References

Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. 2015. Heads-up Limit Hold'em poker is solved. *Science* 347, 6218 (2015), 145–149. `https://science.sciencemag.org/content/347/6218/145`

Noam Brown and Tuomas Sandholm. 2018. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science* 359, 6374 (2018), 418–424. `https://science.sciencemag.org/content/359/6374/418`

Kangxin He, Haolin Wu, Zhuang Wang, and Hui Li. 2022. Finding nash equilibrium for imperfect information games via fictitious play based on local regret minimization. *International Journal of Intelligent Systems* 37, 9 (2022), 6152–6167.

Johannes Heinrich, Marc Lanctot, and David Silver. 2015. Fictitious self-play in extensive-form games. In *International conference on machine learning*. PMLR, 805–813.

Johannes Heinrich and David Silver. 2016. Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. arXiv:1603.01121 [cs.LG] `https://arxiv.org/abs/1603.01121`

Yudong Lu, Youpeng Zhao, Wengang Zhou, Houqiang Li, et al. 2023. Danzero: Mastering guandan game with reinforcement learning. In *2023 IEEE Conference on Games (CoG)*. IEEE, 1–8.

MeriStation. 2024. 'Liar's Bar', el nuevo juego viral de Steam es un mentiroso ... que ha destronado a DiabloIV. *MeriStation (via AS.com)* (2024).

Martin Moravčík, Neil Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. 2017. DeepStack: Expert-level artificial intelligence in heads-up no-limit poker. *Science* 356, 6337 (2017), 508–513. `https://science.sciencemag.org/content/356/6337/508`

Tuo Qu, Qibin Zhou, Jin Zhu, and Fuqing Duan. 2022. Strategy Optimization of Imperfect Information Games Based on NFSP with DDQN. In *International Conference on Guidance, Navigation and Control*. Springer, 4376–4383.

Oskari Tammelin. 2014. Solving Large Imperfect Information Games Using CFR$^+$. *arXiv preprint arXiv:1407.5042* (2014). arXiv:1407.5042

Finbarr Timbers, Nolan Bard, Edward Lockhart, Marc Lanctot, Martin Schmid, Neil Burch, Julian Schrittwieser, Thomas Hubert, and Michael Bowling. 2022. Approximate exploitability: Learning a best response in large games. arXiv:2004.09677 [cs.LG] `https://arxiv.org/abs/2004.09677`

Yifan Yanggong, Hao Pan, and Lei Wang. 2024. Mastering the Game of Guandan with Deep Reinforcement Learning and Behavior Regulating. *arXiv preprint arXiv:2402.13582* (2024).

Martin Zinkevich, Michael Johanson, Michael Bowling, and Carmelo Piccione. 2007. Regret Minimization in Games with Incomplete Information. In *Advances in Neural Information Processing Systems 20 (NeurIPS 2007)*. Curran Associates, Inc., 1729–1736.