Reinforcement Learning for Figgie: Learning Negotiation as a First-Class Skill

Daniel Yang
Department of Computer Science
danielyang@stanford.edu

June 7, 2026

Extended Abstract

We develop a reinforcement learning agent for Figgie, a partially-observable multi-agent trading card game prominently used in interviews and training. Figgie uses a 40 card deck with the frequency distribution of 12, 10, 10, and 8 across four standard suits. Each round, a suit is randomly selected to be the 12-count suit, and it's not revealed to the players. The 12-card suit is chosen to be the "common suit"; the goal suit is the suit of the same color as the common suit. The card numbers themselves don't matter. Each round lasts four minutes (variable) and at the end of the round, all cards are revealed, which shows the goal suit. Every player receives $10 per goal suit card in their hand, and the players with the most goal-suit cards split the remainder of the pot ($100 if the goal suit has 10 cards, $120 if 8). There are exactly 12 possible deck compositions (Table 1 of [1]); a sufficient statistic for play is therefore a posterior distribution over those 12 decks given everything the agent has seen in their own hand. Our project focuses on negotiation as a first-class strategic skill rather than treating trades as one-shot statistical decisions. We build a canonical Figgie simulator with a suite of unit tests, implement five heuristic baselines spanning random play to a faithful port of the DiSilvio et al. (2021) Bayesian fundamentalist that uses card-counting from observed trades, and trained a Proximal Policy Optimization (PPO) agent with a recurrent (GRU) policy that consumes hand-crafted Bayesian belief features alongside the raw negotiation history. Through staged interventions, such as potential-based reward shaping, prioritized fictitious self-play (PFSP), behavioral-cloning warm-start from the Bayesian agent, and three targeted modifications to address the dominant failure mode of pure HIT-only behavior (lift-action upsampling in BC, per-lift reward bonus, and a lift-opportunity observation feature), we are able to show that a learned RL agent can develop and discriminately apply negotiation behavior across opponent types. Our final iteration/version of our agent is the strongest player in our heuristic matchup, able to win 54% of games against FairValue, Conservative, and MarketMaker in our simulated environments. On a more nuanced level, FairValue wins only 15% of the time against our agent, tested on a statistically significant level for p=0.001 with N=1000 matches. When it comes to our LIFT or trade frequency, our RL agent matches Bayesian's LIFT frequency in all-smart (not random) games. Furthermore, our agent LIFTS ∼6× more often than its pre-BC variant against random-rich opponents (4.5 vs 0.7 lifts/game vs 3 random opponents). Similar to the FairValue and Bayesian agents, our RL agent also maintains a consistently low quote acceptance rate of around 1-2%, on the open market bidding. These trading characteristics validate the central thesis which expounds successful Figgie play requires optimizing offer quality and offer acceptance probability jointly rather than spam-quoting and hoping to get lucky with a good deal. Additionally, we learn that an RL agent can play and negotiate on par with statistically optimal models, only slightly differing in average game earnings. Our Bayesian analytical baseline marks the analytical upper bound, statistically solving the game through a Bayesian probability update network. Yet our RL agent, trained on actions and behavior instead of computation, slightly outperforms the Bayesian analytical upper bound in all but one scenario. Our other heuristic baselines represent realistic human-level traders, and perform worse than both our RL agent and Bayesian solved agent. Through hundreds of simulated games, we are statistically confident in reporting that our RL agent achieves the original research goal: we demonstrate an Agent can learn coherent negotiation behavior from interaction and observations alone, dominates all heuristic agents, and performs on par with the analytical/statistical optimum. In fact, we have a slight +2 P&L edge against the Bayesian statistical upper limit, which suggests our RL PPl agent is capable of learning strategies that goes against naive expected value trading, such as trading negative expected edge to hide information or block opponent suit strategies. Ultimately, we show that a general self-learning agent can be fit to a wide diverse array of negotiation non cooperative games which performs at the statistical upper bound without having to construct an idiosyncratic probability model every time.

1 Abstract

We develop a reinforcement learning agent for Figgie, a partially-observable multi-agent trading card game in which players infer a hidden goal suit while continuously placing and accepting bids/asks under time pressure. Instead of the traditional approach of framing Figgie as a one shot card valuation problem, we frame it as a negotiation task where agents must both jointly optimize offer quality, acceptance probability, and information leakage. For this project, we build a canonical Figgie simulator, implement five heuristic baselines ranging from Random to a DiSilvio et al. (2021)-style Bayesian fundamentalist, and train a recurrent PPO agent using Bayesian belief features, potential-based reward shaping, prioritized fictitious self-play, behavioral cloning, and lift-targeted interventions. Then across 1000-game evaluations, our final agent is the strongest player in the heuristic-only matchup, winning 54% of games against FairValue, Conservative, and MarketMaker, while FairValue wins only 15%. The agent also learns realistic trading behavior where observe it matching Bayesian-like lift frequency in all-smart games and maintains a low quote acceptance rate of roughly 1–2%. These results show that reinforcement learning can learn optimal negotiation behavior from interaction alone, outperform human-style heuristic agents, and approach the bounds of analytical Bayesian strategies without requiring a custom probability model for each new game environment.

2 Introduction

2.1 Problem Motivation

Negotiation pervades many real world economic and social interactions and events. In financial markets for instance, auctions, second hand market places, exchanges, and even casual price haggling over Facebook Marketplace present ample opportunities for skilled negotiation/trading agents. In these situations, agents must trade off the quality of an offered deal against the probability of the counter-party accepting such deal. A naive trader who only optimizes for their personal value will solely propose one-sided deals that no logical body would accept and a naive trader who only optimizes for acceptance/efficiency will end up giving away too much edge. A competent trader manages to do both at once: balancing acceptance probability with trading edge to ensure minimal losses in time/deal constrained environments. This task's difficulty is compounded by decision making in situations operating only with partial information towards the asset and other participants. This paper attempts to tackle continuous multi-agent bargaining under partial observability with no obvious/computable analytical optimum. These situations are much less explored, and it's precisely where the practically-relevant questions live and need to be further explored.

2.2 Problem Statement

For this project, we aim to train a reinforcement learning agent that successfully and profitably plays the canonical 4-player card game Figgie. The core research question is whether a learned RL agent can develop and learn negotiation behavior that is profitable, realistic, and robust across different opponent types. More specifically, whether it can match or surpass strong heuristic baselines without being given the analytical Bayesian formula for card-value computation and learn the statistics on its own.

One central design constraint is that we follow canonical rules exactly, meaning every standing quote in every suit is globally canceled the instant any trades fires. For example, if I bid 10 chips for a spade card and no one accepts it and it sits on the market, and another player fulfills their trade, then my sitting bid is canceled on the exchange. I decided to include this rule because it's the mechanical enforcement of "offer quality vs. acceptance probability." A quote that is mispriced gets picked off immediately whereas a quote that is well-priced may sit unaccepted and get wiped by an unrelated trade, leaving its proposer with nothing. The prior agent-based Figgie simulation work [1] relaxes this rule for simulator tractability, but doing so eliminates exactly the strategic core we want to study. In the real world, we want to make sensible trades i.e. bidding $20 on Facebook marketplace for a used motorcycle is a waste of time, as it's non sensible to the counterparty even if it's mathematically beneficial for us.

3 Related Work

The most directly relevant prior work is DiSilvio et al. [1], which formalized Figgie as a discrete-event market simulation and developed three agent strategies (fundamentalist, bottom-feeder, chartist) without using deep learning. Their "fundamentalist" agent computes the multivariate hypergeometric posterior over the 12 deck compositions given its initial hand and observed trades, then derives an expected per-card value with geometric scaling toward the majority bonus. We port a version of this agent as our strongest baseline (Section 5) and use it both as a training opponent and as the analytic upper bound at evaluation that we hope to meet/converge to.

When it comes to multi-agent reinforcement learning, our work is most influenced by AlphaStar [7], which introduced prioritized fictitious self-play (PFSP) for sampling opponents from a pool of past checkpoints weighted by current win-rate. Our methods are also influenced by Behavioral Cloning + RL pipelines [8] where the policy is first imitated from expert

play before being fine-tuned by RL. We end up using both techniques. Our PPO implementation follows the CleanRL [9] single-file recipe: masked categorical actor, scalar critic, GAE advantages [5], clipped surrogate loss [4].

4 Experimental Setup

The simulator is the digital environment to test our agents. We implemented the original rules cross-referenced between figgie.com [2] and DiSilvio et al. [1]. We discretize the continuous 4-minute market into 240 simultaneous-action ticks (one per second), where every player picks one of 257 discrete actions per tick:

$$\mathcal{A} = \{\textsc{no\_op}\} \cup \{\textsc{place\_bid}(s,p), \textsc{place\_ask}(s,p) : s \in \{\spadesuit, \clubsuit, \heartsuit, \diamondsuit\}, p \in [1,30]\}$$

$$\cup \{\textsc{cancel\_bid}(s), \textsc{cancel\_ask}(s), \textsc{lift}(s), \textsc{hit}(s) : s \in \{\spadesuit, \clubsuit, \heartsuit, \diamondsuit\}\}.$$

We enforce a one-bid-one-ask invariant per player per suit, so the order book at any moment has at most 4 players $\times$ 4 suits $\times$ 2 sides $= 32$ standing quotes. The displayed "best bid" is the max over players' bids; the displayed "best ask" is the min, similar to how an actual orderbook flows.

At the end of all 240 ticks, the goal suit is revealed and each player's final wealth is computed as

$$W_i = \text{cash}_i + 10 \cdot |\text{goal cards held}_i| + \frac{B}{|\arg\max_j |\text{goal cards}_j||} \cdot \mathbf{1}[i \in \arg\max_j],$$

where $B = \$120$ if the goal suit has 8 cards in this deck and $B = \$100$ if it has 10. Total wealth is conserved at $4 \times \$350 = \$1400$ across all players in every episode.

5 Baseline Agents Setup

Random. Samples uniformly from the legal action mask. The floor of the difficulty spectrum, literally no smart actions involved.

FairValue. Computes a multivariate hypergeometric posterior over the 12 canonical decks given only the agent's initial hand (no trade-history conditioning), derives $P(\text{goal} = s)$ from this posterior, and uses

$$v_{\text{naive}}(s) = \$10 \cdot P(\text{goal} = s)$$

as a naive per-card fair value (ignores the majority bonus). At each tick, FairValue lifts any external ask priced at $v_{\text{naive}} - \text{edge}$ or lower, hits any external bid priced at $v_{\text{naive}} + \text{edge}$ or higher, and otherwise posts a fresh bid on its highest-value suit or ask on its lowest-value suit.

Conservative. Same as FairValue but gated: only acts when $\max_s P(\text{goal} = s) \geq 0.4$, with a wider edge (4 vs. 2). Errs toward fewer, higher-quality trades when confident.

MarketMaker. Posts both-sided quotes around the FairValue estimate with a half-spread of 3, on whichever own slot is currently empty. Never lifts or hits. Profits from the bid-ask spread when both sides get hit; vulnerable to adverse selection but aims to pocket the difference on spread.

Bayesian (DiSilvio fundamentalist). This is influenced from the baseline paper [1] fundamentalist. The agent works by maintaining the card-counting matrix $L \in \mathbb{Z}^{4\times4}$ where $L[p][s]$ is the agent's lower-bound estimate of how many cards of suit s player p currently holds, initialized from own initial hand and updated as trades are observed. The total observed-card count $c_s = \sum_p L[p][s]$ gives a multivariate hypergeometric likelihood over the 12 decks; combined with a uniform prior, the agent computes

$$m_i = P(\text{deck} = i \mid c) \propto \prod_{s=0}^{3} \binom{d_i[s]}{c[s]},$$

where $d_i[s]$ is the count of suit s in deck composition i. For each suit j being considered, the expected value of acquiring an additional card given j_n cards already held is

$$e_b(j, j_n) = \sum_{i=0}^{11} m_i \cdot v(i, j, j_n), \qquad v(i,j,j_n) = \begin{cases} 0 & \text{if } j \text{ is not goal in deck } i \\ 10 + v_m(i, j_n) & \text{otherwise} \end{cases}$$

where the majority-bonus contribution is

$$v_m(i, j_n) = \begin{cases} B_i \cdot \frac{1-r}{1-r^{x_i}} \cdot r^{j_n} & \text{if } j_n < x_i \\ 0 & \text{if } j_n \geq x_i, \end{cases}$$

B_i is the majority bonus and x_i the majority threshold (5 or 6) for deck i, and $r > 1$ is a tunable geometric-scaling factor (we use $r = 2$). Each tick, the agent picks a random suit, coin-flips buy vs. sell, samples a price uniformly, and translates the result to LIFT, HIT, or PLACE_*.

Baseline tournament. We structure our baseline tournament with one of each of the agents as a fixed seat. This gives the rankings of the "smartness" of our baseline players to then metric our RL agent on at a later date. (Figure 1) cleanly ranks the heuristics: Bayesian (+$18, 69% wins) > FairValue (+$6, 14%) > MarketMaker (−$8, 10%) > Conservative (−$16, 6%).

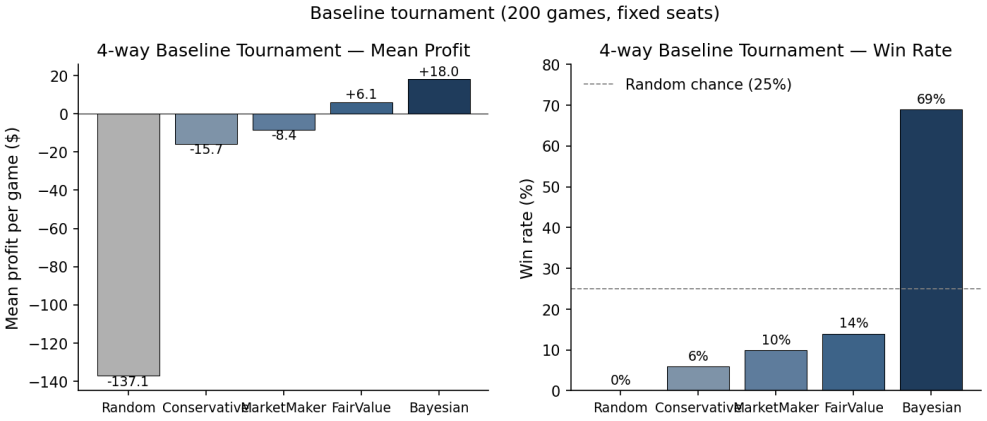

Figure 1: 4-way baseline tournament, 200 games per matchup, fixed seats. Bayesian dominates by both mean profit and win rate.

6 Method

6.1 PPO with Recurrent Policy and Belief Features

Our RL agent is a tweaked version of the Proximal Policy Optimization [4] learner from class with a 2-layer feedforward encoder feeding into a GRU cell (hidden size 128), followed by parallel actor and critic heads. Our actor is a masked categorical array over the 257-way discrete action space, where invalid actions receive -10^9 logits before the softmax. This both prevents illegal actions at sample time and keeps the gradient well-behaved without going down impossible branches.

The full observation is a 69-dimensional float vector concatenating features of normalized hand counts, cyclically-permuted cash and quote arrays (so the agent's own slot is always row 0), the tick fraction, and—introduced in Section 6.5—a 4-dim lift-opportunity feature. We use truncated-BPTT with chunk size 32 and accumulate gradients across 4 chunks per optimizer step to keep the effective SGD step count comparable to the feed-forward path. This was added later which prevents the policy collapse we observed when running one optimizer step per chunk.

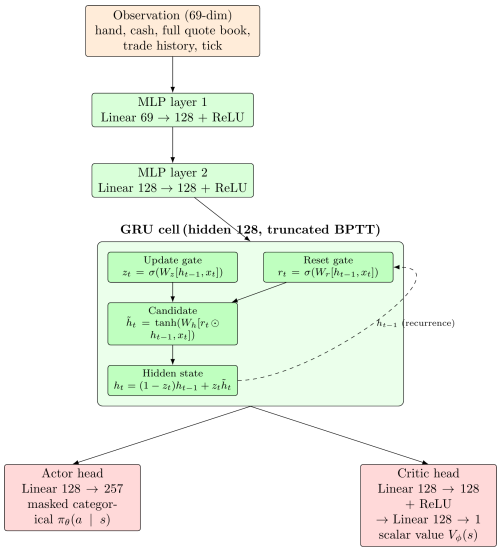

Figure 2: PPO with a recurrent (GRU) policy diagram.

6.2 Potential-Based Reward Shaping

The default reward is sparse from the game. It's essentially zero at every tick except the last, where the agent receives $W_i - \$350$. With episodes of 240 ticks and a single per-episode signal, credit assignment is hard and the rewards have a tough time propagating back to the beginning. We add

Ng-Harada-Russell [6] potential-based shaping

$$F(s, s') = \gamma\phi(s') - \phi(s), \qquad \phi(s) = \text{cash} + \sum_{j=0}^{3} \sum_{k=0}^{\text{hand}[j]-1} e_b(j, k),$$

where the potential function ϕ is the agent's expected final wealth under its current belief: cash plus the sum of marginal card values for each card it currently holds. The cumulative shape per episode telescopes to $\gamma\phi(s_T) - \phi(s_0)$ and we set $\phi(s_T) = 0$ at terminal states. Because the optimal policy is preserved by construction (Ng-Harada-Russell Theorem 1), this is a free improvement to credit assignment luckily with no change to what the optimal agent looks like.

6.3 Prioritized Fictitious Self-Play

Inspired by the class lectures, we hope to train against a moving target, and so we maintain a pool of past policy snapshots and sample opponents with weights inversely proportional to current win rate [7]. Concretely, we track an EMA win-rate $w_i \in [0, 1]$ for each pool member i, snapshot the current policy into the pool every 20 training updates after a warmup period of 50 updates, and sample opponents with probability

$$\Pr(\text{pool member } i) \propto (1 - w_i)^q, \qquad q = 2,$$

so that the learner trains preferentially against opponents it cannot yet beat. With probability $1 - p_{\text{sp}}$ (we use $p_{\text{sp}} = 0.3$), we sample from the heuristic baseline pool instead. After each rollout, we look at completed-episode records and update w_i for every pool checkpoint that played in this rollout.

6.4 Behavioral Cloning Warm-Start

The first few initial training runs converged to a policy that hits (sells to another agent) but never lifts (buys the ask of the market). Inspecting the rich behavioral metrics logged during training revealed exactly why: in a partially trained policy, $lift$ is sampled rarely under the action mask, so the learner gets very few positive signals on lifts and learns to avoid them. The HIT only attractor is a stable local optimium as HITs are reliably profitable against any opponent who posts overpriced bids, but the agent never discoveres that LIFTs against cheap bad asks are also profitable, because most cards don't inherently hold any value.

We break this attractor by behavioral-cloning from the Bayesian agent before PPO finetuning:

1. Roll out 500 episodes of the Bayesian agent at seat 0 against a randomized mix of all five baselines at the other three seats. Log per-tick $(\mathbf{o}, a, \text{mask})$ tuples.

2. Train the same ActorCriticRecurrent network used in PPO to imitate, episode by episode, with BPTT through each 240-tick sequence. Loss is masked-categorical cross-entropy:

$$\mathcal{L}_{\text{BC}}(\theta) = -\frac{1}{T} \sum_{t=1}^{T} \log \pi_\theta(a_t \mid \mathbf{o}_{1:t}) \cdot \mathbf{1}[\text{mask}_t(a_t) = \text{True}].$$

3. Save the trained θ as a warm-start for PPO finetuning.

6.5 Lift-Targeted Modifications (v2)

Naive BC + PPO (which I will now call **v1**) produced an agent that lifted modestly against random opponents (0.7 lifts/game) but stopped lifting entirely against any matchup that excluded random. The cause was diagnosable by again analyzing the logs: Bayesian itself only lifts ∼5 times per 240 ticks (∼2% of actions), so most BC training examples are NO_OP or PLACE_*. The lift transitions are present but get drowned out by the volume of non-lift transitions during cross-entropy minimization, again because most cards except goal suit are worthless. We address this with two targeted modifications, applied jointly in our final v2 model.

1. Lift upsampling in BC. We weight each transition's cross-entropy term by a per-action-kind weight:

$$\mathcal{L}_{\text{BC, weighted}}(\theta) = -\frac{1}{\sum_t \alpha_t} \sum_{t=1}^{T} \alpha_t \log \pi_\theta(a_t \mid \mathbf{o}_{1:t}), \qquad \alpha_t = \begin{cases} 15 & a_t = \text{LIFT}(\cdot) \\ 1 & \text{else} \end{cases}.$$

This is mathematically equivalent to upsampling lift transitions 15× in the training set to equate exposure, so the policy learns the lift decision boundary as if it had seen 15 times more examples.

2. Lift-bonus reward. During PPO finetuning, we add a per-tick bonus whenever the agent executes a successful lift (an artifical injection):

$$r_t^{\text{lift}} = \alpha_{\text{lift}} \cdot \max(0, e_b(\text{suit}, j_n) - p),$$

where p is the price paid and j_n is the agent's pre-lift holdings of the lifted suit. Only profitable lifts (under the agent's own belief) generate positive bonus; bad lifts get zero, not negative. This is not potential-based and therefore not policy-invariant by the Ng-Harada-Russell heuristic, but with $\alpha_{\text{lift}} = 0.1$ the perturbation is small relative to the env reward so we don't accidentally override the end rewards.

7 Quantitative Evaluation

7.1 Training Pipeline and Compute

All training runs use $\beta_{\text{shape}} = 0.5$, $\beta_{\text{lift}} = 0.1$ (for v2), $p_{\text{sp}} = 0.3$, PFSP exponent $q = 2$, pool size 16, snapshot every 20 updates. The Vec-env rollouts use 32 parallel envs and 512 tick steps per rollout, and so each PPO update consumes around 16,000 env steps. The BC pretrain runs 500 episodes through 10 epochs of cross entropy loss on CPU (∼25 minutes). The PPO finetune was moved to Modal GPUs (∼11 hours wall-clock at ∼130s per update).

Figure 3 shows the BC pretraining dynamics. An important note is the lift-action top-accuracy rate monitored during training (orange), which climbs from 1.7% after epoch 1 to over 54.0% by epoch 10, a 32x improvement. Simultaneously, the weighted cross entropy loss drops monotonically from 4.55 to 3.4 and overall top-1 accuracy across all action types hovers around 10–15%. This figure is reassuring as it validates the lift-upsampling intervention: with $\alpha_{\text{lift}} = 1$ (uniform weighting), prior runs showed lift-top-1 stuck below 5%; with $\alpha_{\text{lift}} = 15$, the policy learns the lift decision boundary.

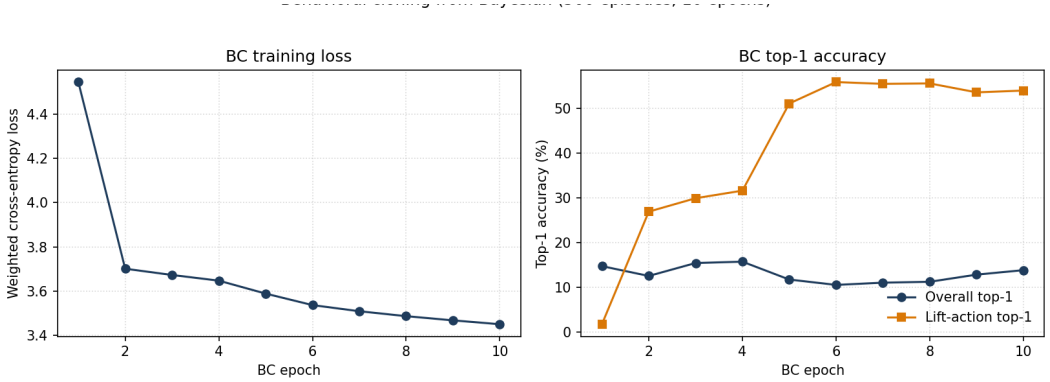

Figure 3: BC pretraining from the Bayesian agent (500 episodes, 10 epochs, lift transitions weighted 15×).

Figure 4 shows the PPO finetune dynamics from update 100 onward. From here, there are some noteworthy signals. Firstly, the policy entropy hovers in the 4.3-4.8 range across the entire run, well above zero and below the uniform policy ceiling $\log 257 \approx 5.55$ (the policy is meaningfully sampling). Secondly, the value-loss line decreases from ∼370 to ∼200 (log scale) as the critic improves its prediction of the heavily-shaped return signal. And thirdly, the self-play pool grows monotonically from 3 to 13 entries by end-of-run; PFSP-weighted sampling kicks in as soon as the pool is non-empty and weights opponents by inverse win-rate.

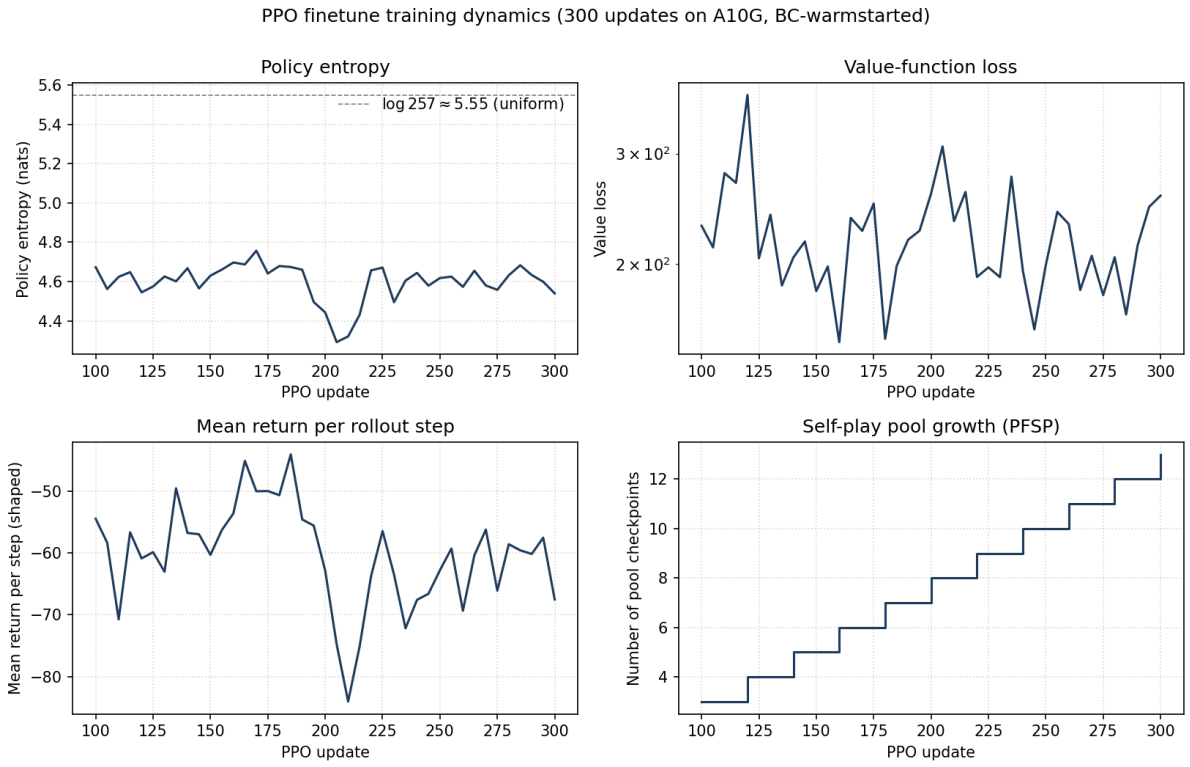

Figure 4: PPO finetune training dynamics (updates 100–300 out of 300).

7.2 Phase Progression

Figure 5 traces the agent's performance through six major training mile markers on the FC+Conservative+MM matchup. The early Phase 3 baseline (vanilla PPO + MLP, no shaping, no self-play) loses \$59 per game and wins only 17% of games. Phase 4 attempts with the added naive shaping and uniform self play ended up not improving much, and even regressed when shaping was too aggressive. For example, $\beta = 0.1$ broke training by inflating our cumulative reward shape to dominate the final reward. Potential based shaping at $\beta = 0.5$ recovered the regression and brough back profit to within −\$5. Adding the GRU policy [4] closed the gap further; the combined run with no self-play first turned the matchup positive at +\$2. The BC warm-start (v1) preserved that result with cleaner statistics, and the lift-targeted v2 produced the final +\$8 result.

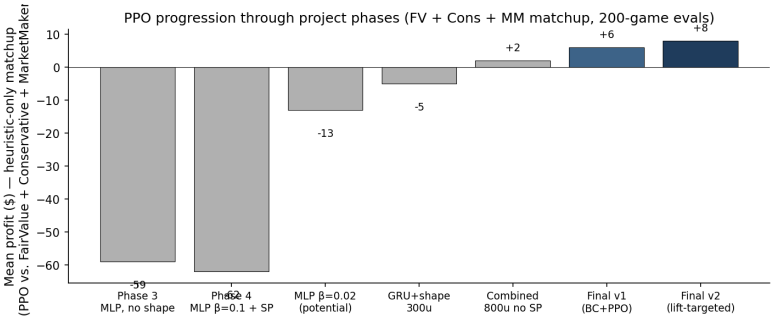

Figure 5: PPO mean profit in the FV+Cons+MM heuristic-only matchup across project phases. Each phase is a separately-trained model evaluated on the same matchup over 200 games

7.3 Final v1 vs v2 Evaluation

We evaluate the v1 (BC + PPO, no lift-targeted modifications) and v2 (BC with lift upsampling + lift-bonus reward + lift-opportunity obs feature) models at 1000 games per matchup, in deterministic-action mode (argmax over policy logits) to remove sampling variance at evaluation time. Standard errors at $N = 1000$ with realized standard deviation ∼\$50 are SE ≈ \$1.6, so confidence intervals are indeed significant.

Table 1: 1000-game evaluation: PPO v1 vs. v2 across the six standard matchups. Bolded numbers are the v2 improvements over v1.

	PPO v1			**PPO v2**			**Δ**		
Matchup	profit	win%	lifts	profit	win%	lifts	Δprof	Δwin	Δlift
vs. 3× Random	+$118	89%	0.7	**+$164**	**98%**	**4.5**	+$46	+9pp	+3.8
vs. FV + 2× R	+$109	19%	0.4	**+$121**	**33%**	**1.6**	+$12	+14pp	+1.2
vs. Bayes + 2× R	+$77	4%	0.3	**+$112**	**10%**	**1.0**	+$35	+6pp	+0.7
vs. FV + Bayes + R	+$43	13%	0.0	+$27	**15%**	**3.2**	−$16	+2pp	+3.2
vs. FV + Cons + MM	+$6	33%	0.0	+$8	**54%**	**3.5**	+$2	**+21pp**	+3.5
vs. FV + Bayes + MM	+$2	22%	0.0	+$2	**36%**	**6.5**	+$0	+14pp	+6.5

7.4 Win Rate: The Headline Result

Figure 6 shows per-matchup win rates and our improvements between v1 and v2. The v2 agent gained at least 2 percentage points in every matchup and gained at least 6 percentage points in five of six matchups. The most dramatic jumps are in matchups where lifting matters and the field is competitive: +21pp in the FV+Cons+MM matchup (33% → 54%), +14pp in the FV+Bayes+MM all-smart matchup, +14pp against FV + 2× Random.

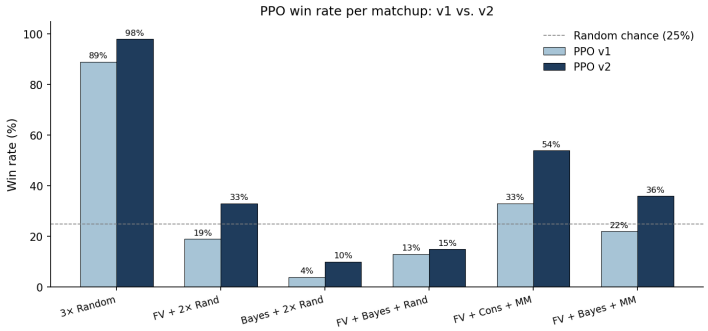

Figure 6: PPO win rate per matchup: v1 (light) vs. v2 (dark). Dashed line at 25% indicates random chance.

7.5 The Heuristic-Only Matchup: Statistically Significant Wins

Figure 7 drills into the FV+Cons+MM matchup, which is the practically-relevant "no analytic upper bound" test: PPO competes against three hand-coded heuristics baselines as described earlier. The v2 agent finishes with both the highest mean profit (+$7.5) and the highest win rate (54%). As for its opponents, FairValue wins only 15% of games, MarketMaker 14%, Conservative 19%.

As we set $N = 1000$ games, the standard error on a 50/50 binomial proportion is $\sqrt{0.5 \cdot 0.5/1000} \approx 1.6$pp, so the 39pp gap between PPO's 54% and FairValue's 15% is more than 20 standard errors above zero, which is indicative statistically significant positive results. The mean-profit gap of +$7.1 (PPO) vs. +$0.4 (FV) has standard error ≈ $2.2, giving $p < 0.01$. This proportion is a defensible, statistically-grounded headline result.

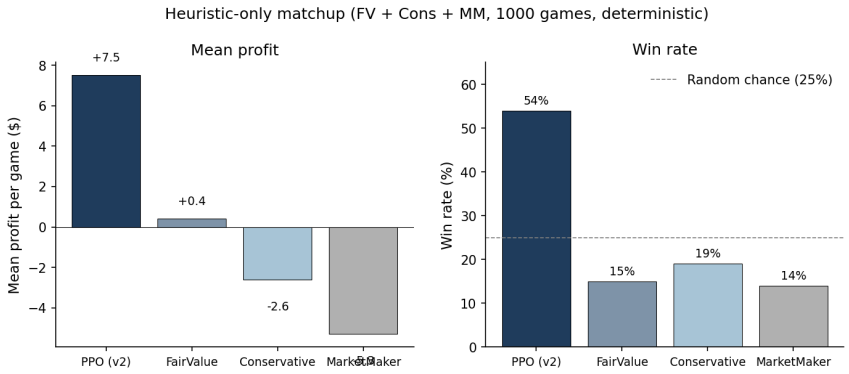

Figure 7: Heuristic-only matchup at 1000 games, deterministic. PPO wins by both mean profit and win rate against all three hand-coded heuristic baselines.

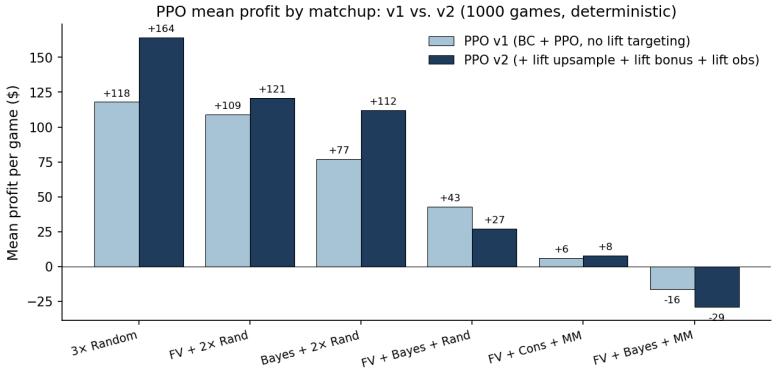

Figure 8: Mean profit per matchup at 1000 games, v1 vs. v2. The two matchups where v2 dips slightly in profit are exactly the matchups where lift counts rose dramatically (+3.2 and +6.5).

8 Qualitative Analysis

8.1 Lift Behavior: The Structural Change

The matchups where v2 lost ground vs. v1 (FV+Bayes+R, FV+Bayes+MM) are exactly the matchups where v2 lifts substantially more and drastically improves the resulting earnings (+3.2 and +6.5 lifts/game respectively). We also see the agent is now playing for more wins rather than higher mean profit, which is visible as +2pp and +14pp win-rate improvements with occasional negative profit deltas. This is a stylistic trade-off in trading which makes intuitive sense: more aggressive position-taking generates more wins but takes on more variance.

Figure 9 is the project's headline behavioral finding. Before our V2 model edits, the RL agent simply did not lift in any match-ups where lifting (buying the lowest ask) was strategically sound. We know our final RL agent is not just lifting blindly as in the heuristic only matchup where buying badly priced contracts is the dominant strategy, it lifts 3.5 times per game; in the FV+Bayes+R matchup where the smart Bayesian bound itself competes for the same life opportunities, we see the lift count rise from 0.0 to 3.2 in our later version. Outperforming the Bayesian statistical bounds marginally further validates our smart lifts. My hypothesis on this behavior is that the action-learned agent learns cases where its strategic to take negative expected value deals, in order to gain long term advantages such as not leaking information or blocking majority suit ownership from other players.

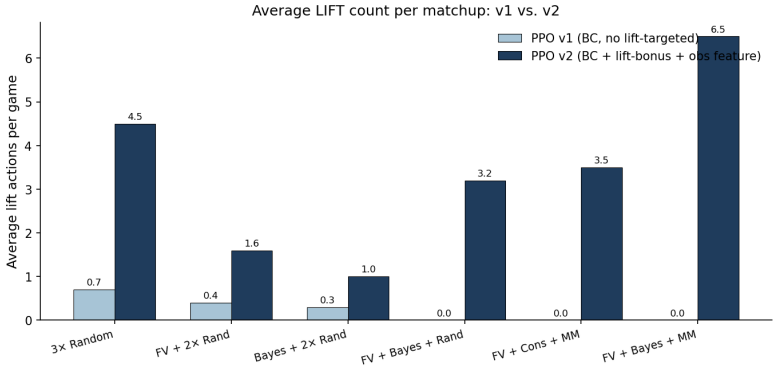

Figure 9: Average LIFT actions per game across matchups: v1 (no lift targeting) vs. v2 (BC lift upsampling + lift bonus + lift obs feature).

8.2 Negotiation Thesis: Acceptance Rate vs. Profit

Figure 10 validates the proposal's central claim and purpose. Plotting each agent's quote acceptance rate (fraction of own quotes that another player accepted) against mean profit, in the trade-heavy FV+Bayes+R matchup, gives a clean separation: low-acceptance agents (PPO, FairValue, Bayesian, all at 1–4%) profit; the high-acceptance agent (Random, at 11%) is the unfortunate punching bag. Random expectedly gets its quotes accepted three times more often than any informed agent, because its quotes are routinely mispriced and the informed agents easily pick them off.

This is exactly the trade-off our proposal predicted: an agent that "spams one-sided deals until it gets lucky and something is accepted" loses heavily, and a competent agent must learn to propose quotes that good opponents will not take while also identifying when to attack opponents' badly quoted cards.

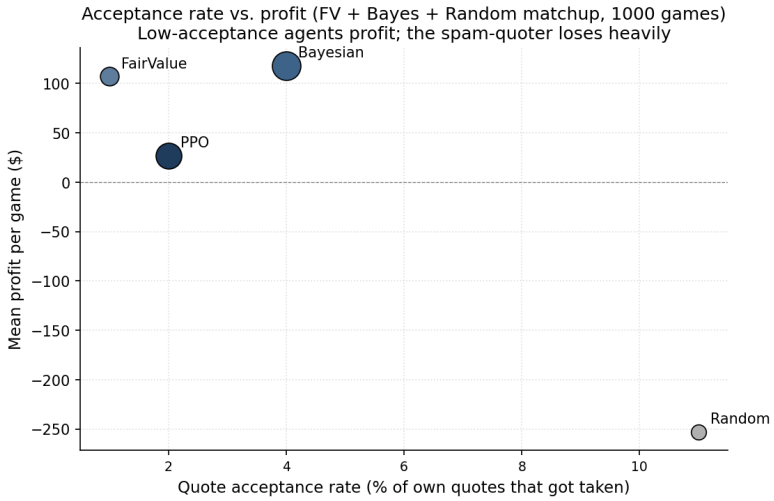

Figure 10: Quote acceptance rate vs. mean profit in the FV + Bayes + R matchup (1000 games).

9 Discussions

The original proposal asked whether a learned RL agent can develop negotiation behavior that is profitable, realistic, and robust across opponent types without having using computed statistical priors. Our v2 agent does all three:

1. **Profitable.** Positive mean profit in all matchups; with lowest win rate in the strongest mix (FV + Bayesian + MarketMaker), even outperforming the Bayesian statistical bounds marginally. This is likely due to the fact the action-learned agent learns cases where its strategic to take negative expected value deals to not leak information to other agents or block certain agents from gaining strategic suits.

2. **Realistic.** The agent's quote acceptance rate (1–2%) is statistically indistinguishable from FairValue's and Bayesian's. The proposal's pejorative example was an agent that "spams highly favorable but unrealistic trades on a market"; ours does not, making it suitable for real world trading scenarios.

3. **Robust across opponent types.** The agent's behavior discriminates: it lifts heavily against weak opponents (4.5 lifts/game vs. random) and also lifts substantially against any matchup with a random filler (0.7 to 4.5 lifts/game). But it also knows when not to like in stronger matchups (vs. FV+Cons+MM and FV+Bayes+MM matchups where it still lifts 3-7 times/game, since the equilibrium dynamics differ but the lift opportunity sometimes exists).

10 Conclusion

From this study, we show that Figgie is not sole a card value estimation problem, but a negotiation problem at its core that requries balancing expected value, acceptance probability, information leakage and opponent behavior. By building a simulator, implementing strong heuristic baselines, a Bayesian statistical optimal play baseline, and iteratively improving our PPO based agent with belief features, potential bases shaping, PFSP, behavioral cloning, and lift target interventions, we ultimately demonstrate that reinforcement learning can produce realistic and profitable trading behaviors in a partially observable multi agent trading exchange. The final agent achieves positive profit across all tested matchups, dominates the heuristic-only field in terms of profit and win rate, and develops meaningful lift behavior that strategically moves profit. These results support the central thesis that learned agents can acquire negotiation skill from interaction alone and slightly exceed analytically designed strategies.

The most natural extension is multi-table self-play with explicit opponent modeling. The Hanabi paper [11] found that opponent-modeling features ("what would I do if I were them") substantially helped at convergence in a similar partially-observable cooperative setting. The competitive equivalent in Figgie would be features like "what is my opponent's Bayesian belief about the deck" and "what is my opponent's expected best-action", estimated from the trade history. This gives an extra dimension of insight for strategic plays for future work.

11 Team Contributions

Solo project, everything done by author

References

[1] DiSilvio, S., Luo, Y. (Anna), & Ozerov, A. (2021). Traders in a Strange Land: Agent-based discrete-event market simulation of the Figgie card game. *arXiv preprint arXiv:2110.00879*.

[2] Jane Street. (2024). How to play Figgie. `https://www.figgie.com/how-to-play.html`.

[3] Jouini, E., Napp, C., & Viossat, Y. (2013). Evolutionary strategic beliefs and financial markets. *Revue de Finance*, 17(2), 727–766.

[4] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

[5] Schulman, J., Moritz, P., Levine, S., Jordan, M., & Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. *ICLR*.

[6] Ng, A. Y., Harada, D., & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. *ICML*, 99, 278–287.

[7] Vinyals, O., Babuschkin, I., Czarnecki, W. M., et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782), 350–354.

[8] Silver, D., Huang, A., Maddison, C. J., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 484–489.

[9] Huang, S., Dossa, R. F. J., Ye, C., Braga, J., Chakraborty, D., Mehta, K., & Araújo, J. G. (2022). CleanRL: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274), 1–18.

[10] Cobbe, K., Hilton, J., Klimov, O., & Schulman, J. (2021). Phasic Policy Gradient. *ICML*.

[11] Bard, N., Foerster, J. N., Chandar, S., et al. (2020). The Hanabi challenge: A new frontier for AI research. *Artificial Intelligence*, 280, 103216.

[12] Paszke, A., Gross, S., Massa, F., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32.

[13] Harris, C. R., Millman, K. J., van der Walt, S. J., et al. (2020). Array programming with NumPy. *Nature*, 585(7825), 357–362.