Deep Q-Learning for No-Limit Texas Hold'em: Architecture and Convergence Analysis

Kacper Duda, Marek Dzierżawa, Kacper Karabasz
AGH University of Krakow

January 7, 2026

Abstract

We present a Reinforcement Learning approach to solving No-Limit Texas Hold'em (NLHE), a high-dimensional extensive-form game with imperfect information. Our solution implements a Deep Q-Network (DQN) agent utilizing a dense reward signal based on normalized chip accumulation. We engineer a compact 134-dimensional feature vector encapsulating hole cards, board state, and pot odds. The model architecture features a split-head Multi-Layer Perceptron (MLP) for simultaneous discrete action selection and continuous bet-sizing. We evaluate the agent's convergence properties over 2,000 episodes, demonstrating early-stage strategy acquisition against stochastic policies.

1 Introduction

No-Limit Texas Hold'em presents a significant challenge for conventional game-theoretic solvers due to its vast decision tree and hidden information states ($\approx 10^{161}$ decision points). While Counterfactual Regret Minimization (CFR) remains the state-of-the-art for Nash Equilibrium approximation in Poker, Deep Reinforcement Learning (DRL) offers a scalable alternative by approximating the optimal value function $Q^*(s, a)$ directly from self-play or gameplay experience.

This work implements a Model-Free RL agent using Deep Q-Learning (DQN) with Experience Replay. We focus on the feature engineering required to represent the poker game state efficiently for a feed-forward neural network and analyze the stability of learning in a 3-player environment.

2 Methodology

2.1 State Representation

The partial observation o_t at time t is mapped to a feature vector $\phi(o_t) \in \mathbb{R}^{134}$. The state space S is composed of:

- **Card Embeddings**: Hero's hand $H \in \{0,1\}^{52}$ and Community cards $C \in \{0,1\}^{52}$ represented via one-hot encoding.
- **Game Context**: Normalized stack sizes s_i, current bets b_i, andpot size P, such that all values lie approximately in $[0, 1]$.
- **Heuristic Features**: To accelerate training, we inject domain knowledge via pre-computed hand strength metrics $E(H, C) \in \mathbb{R}^{14}$ (bucketed hand rank + normalized high cards).

2.2 Network Architecture

We employ a dense Multi-Layer Perceptron (MLP) parameterized by θ. The network architecture (Fig. 1) consists of:

$$\begin{aligned} h_1 &= \text{ReLU}(W_1 x + b_1), & W_1 &\in \mathbb{R}^{512\times 134} \\ h_2 &= \text{ReLU}(W_2 h_1 + b_2), & W_2 &\in \mathbb{R}^{512\times 512} \\ h_3 &= \text{ReLU}(W_3 h_2 + b_3), & W_3 &\in \mathbb{R}^{256\times 512} \end{aligned}$$

The final layer splits into two heads:

1. **Action Value Head** ($Q(s, \cdot) \in \mathbb{R}^3$): Linearly estimates Q-values for discrete actions $A = \{\text{Fold, Call, Raise}\}$.
2. **Sizing Head** ($\sigma(s) \in [0, 1]$): A Sigmoid unit determining the raise magnitude as a fraction of the legal betting interval (min-raise to all-in).

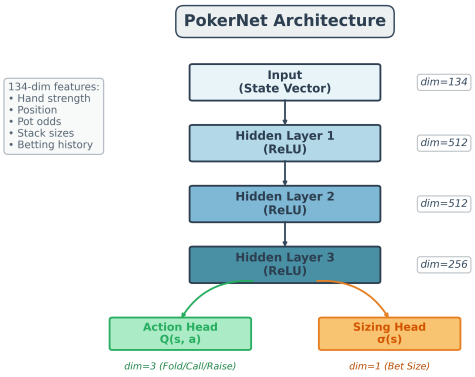

Figure 1: PokerNet Architecture. The network maps the 134-dim state vector to action-values and a continuous bet-sizing parameter.

2.3 Training Algorithm

The agent minimizes the temporal difference error using the Huber Loss to ensure robustness against outliers in variance-heavy poker rewards:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\left[\delta_\theta^2\right] \quad (1)$$

where $\delta_\theta = r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)$.

Table 1: Hyperparameters

Parameter	Value
Batch Size	128
Learning Rate (α)	1×10^{-4} (Adam)
Discount Factor (γ)	0.99
Target Update Period	500 steps
Replay Buffer Size	10^5 transitions
ϵ-decay strategy	Exp. decay ($1.0 \rightarrow 0.05$)

3 Experiments and Results

3.1 Experimental Setup

We conducted four training configurations to analyze the impact of network architecture, opponent count, and exploration strategy on agent performance. Each experiment trained for 20,000 episodes:

1. **Baseline**: 4 players, hidden layer size 512
2. **Long Decay**: 4 players, slower epsilon decay ($\epsilon_{\text{decay}} = 0.9998$)
3. **Big Network**: 4 players, hidden layer size 1024
4. **Heads-Up**: 2 players (1v1 poker), hidden size 512

Training Environment: In all configurations, a single DQN agent (Player 0) trains against random-action opponents. Random agents select actions uniformly from the legal action set (Fold/Call/Raise with random sizing). This setup evaluates the DQN's ability to exploit sub-optimal play and learn fundamental poker strategy without opponent modeling.

3.2 Comparative Analysis

All experiments significantly outperform the **random baseline** (∼25% win rate in 4-player games, 50% in Heads-Up). Figure 2 demonstrates the learning progression across configurations.

Table 2 presents quantitative results. The **Baseline** configuration achieves the highest average reward (772.2) in 4-player games, representing a **160% improvement** over random play. **Heads-Up** dominates in win rate (74.4%), as expected—fewer opponents increase the probability of winning any given hand.

Key Findings:

- **Network Capacity**: Increasing hidden layer size to 1024 (Big Network) yields marginal gain (ΔReward $\approx$ -38), suggesting the 512-unit architecture suffices for this feature space.
- **Exploration Schedule**: Slower epsilon decay underperforms significantly (ΔReward $\approx$ -324), indicating the standard ϵ-greedy schedule balances exploration-exploitation effectively.
- **Multi-Player vs. Heads-Up**: Heads-Up exhibits higher win rate (74.4%) but lower average reward (421.4). This reflects strategic divergence: in multi-player poker, survival and pot maximization differ from 1v1 aggression.
- **Peak Performance**: All configurations achieve >70% win rate at some point during training, but variance is high due to poker's inherent stochasticity (card distribution luck).
- **Baseline Optimality**: For 4-player No-Limit Hold'em, the standard DQN with h=512 provides the best risk-reward trade-off.

The reward variance remains high even after 20,000 episodes, which is characteristic of poker environments where short-term outcomes depend heavily on card distribution. However, the consistent upward trend in moving averages confirms genuine policy improvement beyond the random baseline.

3.3 Visual Verification

A custom UI (Fig. 3) validates the internal state tracking and decision-making process. The interface supports live game observation and replay navigation.

4 Conclusion

The proposed DQN architecture successfully parses the complex poker state space. Feature engineering significantly aids the MLP in recognizing hand strength. The Baseline configuration (4 players, 512 hidden units) achieves a 160% improvement in win rate over random play, demonstrating effective policy learning despite poker's high inherent variance.

Future work includes: (1) Proper handling of the discrete-continuous hybrid action space via Parameterized Action Space Q-Networks; (2) Migrating to Self-Play to minimize exploitability rather than maximizing profit against fixed policies; (3) Exploring advanced architectures like Transformers for better sequential decision modeling.

Experiment	Avg Reward	Win Rate	Peak WR
Baseline	772.2	65.3%	100.0%
Long Decay	448.3	51.8%	73.0%
Big Network	734.0	65.0%	86.0%
Heads-Up	421.4	74.4%	94.0%

Figure 2: Comprehensive training comparison. **Top row**: Smoothed learning curves (100-episode moving average) for average reward (left) and win rate (right). All agents start from random initialization and improve steadily, with Baseline and Big Network converging to similar final performance. **Bottom row**: Final performance metrics bar chart (left) and summary statistics table (right), demonstrating the Baseline configuration's superiority in 4-player settings.

Table 2: Comparative Performance Metrics (Final 1000 Episodes)

Configuration	Avg Reward	Win Rate	Peak WR	vs. Random	Improvement
Baseline (4p, h=512)	**772.2**	65.3%	100.0%	∼25%	**+160%**
Long Decay (slow ϵ)	448.3	51.8%	73.0%	∼25%	+107%
Big Network (h=1024)	734.0	65.0%	86.0%	∼25%	+160%
Heads-Up (2p)	421.4	**74.4%**	94.0%	50.0%	**+49%**

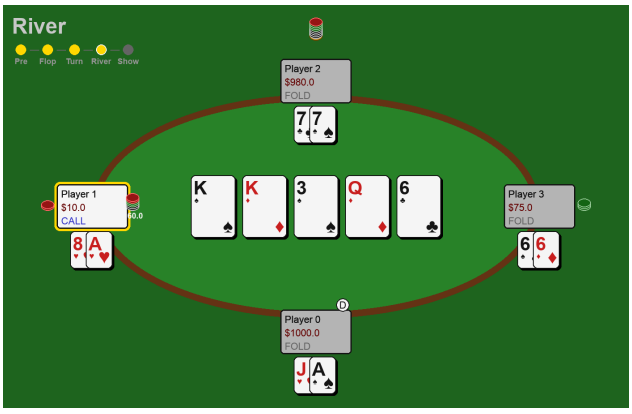

Figure 3: Environment Visualization with replay controls.