Minimax Q-Learning in a Partially Observable Environment

Jacob Kruse and Lingze Zeng

December 8, 2025

Original Proposal

- Begin with the Minimax Q-Learning framework and use it as the foundation of our approach
- Extend Minimax Q-Learning using concepts from POMDPs and extend to POMGs
- Employ the Blackjack environment from OpenAI Gym as the experimental domain
- Modify the standard Blackjack rules to create a two-player, turn-based, zero-sum game

Deviations

- Find a way to extend POMGs for partial observed Minimax-Q, but the performance is not as expected
- Find an alternative way to handle partial observability
  - Assuming dealer's state can be fully observed (One face up card is observed, and that's the state we only care about)

Response to Feedback

- Add fixed policy
- Lack of citations in related works
  - Littman (1994) – Proposed Minimax-Q Learning for two-player zero-sum Markov games under full observability
  - Watkins & Dayan (1992) – Introduced Q-learning, foundation for model-free RL but limited to single-agent MDPs
  - Littman (2001) - Friend-or-Foe Q-learning in General-Sum Games
  - Hu & Wellman (2003) – Nash Q-learning for general-sum stochastic games
- Minimax-Q is incomplete in Markov Games
  - α:Learning rate
  - γ: Discount rate
  - β: Win rate estimation wight

$$Q\ (S,a_i,a_j) = (1\ -\ \alpha) * Q(S,a_i,a_j) + \alpha[r\ +\ \gamma V(S')\ +\ \beta(Estimator(S,a_i)\ - 0.5)]$$

$$V(S') = max_{a_i'} min_{a_j'} Q(S',a_i',,a_j')$$

Empirical Performance Estimator for Improved Reward Evaluation

- Restore win counter in table W with index [obs, a]
    - Obs: Self card sum, Opponent’s faced up card, Useable Ace
    - Action: Stick & hit

Our Blackjack Game

Two-Players
Player and *Dealer*

Turn-Based
The first turn is randomly assigned to the *Player* or *Dealer*
After choosing *Hit* or *Stand*, the choice passes to other player
If a player chooses *Stand*, the other player can *Hit* repeatedly
Game ends when both *Stand* or someone *busts*

Partial Observability
Opposite player only shows one card

Zero-Sum
Win = +1, Loss = -1, Draw = 0

Blackjack Demo

Experiment Setting

- Q Learning vs Fixed policy
- Q Learning vs Q Learning
- Minimax-Q vs Fixed policy
- Minimax-Q vs Minimax-Q
- Minimax-Q vs Q Learning
- Partial Observed MiniMax-Q vs Fixed policy
- Partial Observed MiniMax-Q vs Q Learning
- Partial Observed MiniMax-Q vs Minimax-Q

Results

Trained algorithms for 1,000,000 learning steps and assessed them over 100,000 games

	MMQ vs Fixed	MMQ vs MMQ	POMMQ vs Fixed	MMQ vs POMMQ	MMQ vs Q	POMMQ vs Q	Q vs Fixed	Q vs Q
Wins	44403	44306	21122	77222	45542	18584	44361	44287
Losses	46138	45298	77792	19301	44527	78442	47146	45775
Draws	9459	10396	1086	3477	9931	2974	8493	9938
Win %	0.44403	0.44306	0.21122	0.77222	0.45542	0.18584	0.44361	0.44287
Loss %	0.46138	0.45298	0.77792	0.19301	0.44527	0.78442	0.47146	0.45775
Differential	-1735	-992	-56670	57921	1015	-59858	-2785	-1488

Analysis

Fixed policy performed the best

- Least dependency on state

Agents struggled against fixed policy

- Large state space
  - Player sum [2-31]
  - Dealer card [1-10]
  - Usable ace
- Action dependency
- Different outcomes for same (s,a)

Vanilla MMQ best Agent

- Designed for this scenario

MMQ and Q nearly equal

- Large uncertainty inherent to Blackjack

POMMQ performed the worst

- More on this in next slide

Discussion

The win rate estimator drastically makes the performance worse

- Noisy in early training
- Initial bias in the success-rate estimator
  - success_rate(s,a) - 0.5 = -0.5
- Disrupts convergence

Questions?