Use Reinforcement Learning to Play Heads-up Limit Texas Hold'em

Che Mingzhou, Huang Xuhui, Liu Yichen, Sun Tianbo

Master of Computing Students(Artificial Intelligence), National University of Singapore

Abstract

We use reinforcement learning to train poker agents that can play heads-up limit Texas Hold'em. We first develop a rule-based poker agent that is genuinely intelligent in the game. This rule-based agent plays exactly according to how strong its hand is, with small randomness to avoid being exploited. We then experiment variants of deep neural network (DQN), including double DQN, Dueling DQN, Noisy DQN Neural Fictitious Self Play. Even our rule-based agent is genuinely intelligent, the best variant of our reinforcement learning agent can beat the rule-based agent by small margin in simulation. This suggest that the agent is about to pick up the probabilistic and strategic aspect of Texas Hold'em.

Heads-up Limit Texas Hold'em and Rule-based Agent

At Texas Hold'em, there are many games in one session. At each game, will be total of four rounds of action. At each round, players take turns to act by either "check", "raise", "call" or "fold". A player who folds will exit the current game and get nothing. A player who checks do not need to put in any chips. A player who raises needs to put in chips to the board; after one player raises, all the other player can either fold (exit the current game), call (by putting in the same amount of chips as the player who raised) or re-raise (by putting more chips than the player who raised). Each round only ends if all the players either fold or put in the same amount of chips to the board (either 'all check', or 'raise – call'). At each game, each player is first dealt two cards that cannot be seen by other players. After the first round of action, three cards are dealt to the board that can be seen by all players, called 'flop'. Another two cards would be dealt separately after the second and third round of action completed, called 'turn' and 'river'. Fourth round of action happens after the 5th card is dealt. After the fourth round, all players would show their 2 private cards, combined to the 5 board games, and get the best 5-out-of-7 combination. The winner is the player with the best combination, who can take all the chips from the table. Then a new game will start. Heads-up Limit Texas Hold'em is a variant of Texas Hold'em with only two players, and players can only raise by a fixed amount and re-raise by fixed number of rounds.

(Above) A typical game board

(Right) Common representation of pre-flop hand value

We develop a rule-based agent that play primarily according to the strength of its hand. Before board cards are dealt, and strength of its hard is only a function of two private cards. For each combination, we compute a numeric value that represents its probability of winning, if played against all other combination without knowing any board cards. For example, a hand with two Aces has a value of 1, a hand with off-suit 3 and 2 has value of 0. After the board cards are dealt, we can compute the strength of hand in conjuncture of the board cards. Specifically, we simulate all possible combination of remaining board cards, and calculate the probability of our hand winning against a reasonable range of opponent's hand. We assume opponent playing full range at flop, top 50% range at turn and top 35% at the river. The higher the strength is, the more aggressive the agent is.

Moreover, the rule-based agent would also act randomly with a small percentage. This is a popular 'mixing strategy' adopted by many poker players, primarily to avoid being exploited by other players. For example, if a player is known to only raise with good hand and check with bad hand, once this player checks, other players would know that his or her hand is not good and win by bluffing (raise even if other players have bad hands).

Our rule-based agent is genuinely intelligent, because it incorporates both the statistical aspect (by calculating winning probabilities against other hand range) and the strategic aspect (by strategy mixing). This serves as a good opponent when training reinforcement agents.

State Representation and Performance Measurement

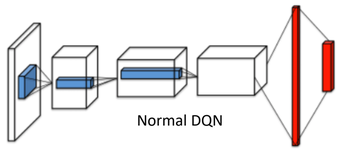
We represent each state by a vector of length 72. The first 52 elements correspond to the 52 different cards in a deck, taking the value of 1 if a card is dealt at either the board or the hand of each player. In other words, different players will see different state at the same game. The last 20 elements represent the betting history. More precisely, the number of raises at each of the four rounds. In Texas Hold'em, it is important to understand how aggressive other players are; number of raises is a good estimate of aggressiveness.

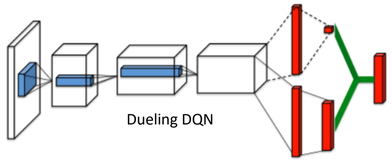
We measure performance of agents based on average big blinds per hand (BBPH) ahead in a 'tournament'. In Limit Texas Hold'em, players can only raise by fixed multiply of big blinds each time. Hence BBPH is good measure of the payoff in a single game. A 'tournament' simply refers to a simulation of multiply games; this is to remove the 'luck' component in the game.

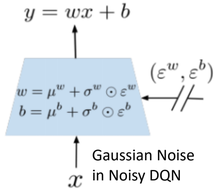
Methodology

Our default model is a double DQN model. We experiment two improvements over this baseline model. We also experimented Neural Fictitious Self Play which also uses neural network in its implementation.

Dueling DQN is a variant of DQN. It has a network structure change in the last layer comparing with the DQN. Dueling DQN explicitly separates the representation of state values and state dependent action advantages. In some of the games, the values of the different actions are similar, and it is less important which action to take. This may cause a slower learning of the Q value in DQN as it update Q value for the state with specific action. In Dueling DQN, as the state and action are split into two streams, it can learn the states without learn the value of each action in each state, so the learning may be faster.







We also implement Noisy DQN, where we utilises noisy linear layers for exploration, instead of ε-greedy exploration in the normal DQN. In our experiment, we apply the factorized gaussian noise (parameterized by mu and sigma) to DDQN model in a fully-connected linear layer to implement the Noisy DQN model.

Fictitious Self Play is a method that fixes the problem of the Fictitious Play by integrating time/sequence by using Extensive Form Game. It also uses Reinforcement Learning to find an approximation of the best response and Supervised Learning to update the average strategy. It is proven that it can converge to a Nash Equilibrium. NFSP built upon classic Fictitious Self Play with a DQN backbone model, and finally achieved what FSP is designed for.

Experiment Design

In our experiment, we have

- 2 benchmark agents: rule-based agent (as we described earlier) and random agent (take random action with equal probabilities).
- 4 different neural-network-based models: DDQN, dueling DQN, Noisy DQN and NFSP.
- This gives us total of 8 different combination of (bench-mark model, training model)

In heads-up Texas Hold'em, there are only two players. We will use a combination of benchmark agent and a neural-network-based model to play against each other. We can see if our neural network model can outperform the benchmark in this heads-up setup.

Specifically for each combination, we repeat the game for 5,000 epoch.

- Initialize random weights of neural network at the first epoch.
- Record the actions and pay-off to replay buffer at each epoch.
- Regularly fit the replay buffer to training model and perform gradient descend; clear the buffer.
- Run a tournament of 5,000 games every 200 epoch, measure the performance.
- After the last epoch, run a tournament of 5,000 games between benchmark model and training model

Finally, for the same training model, we run a tournament of 5,000 games between the one trained using rule-based agent vs. trained using random agent. This will show if there any improvement via the use of rule-based agent.

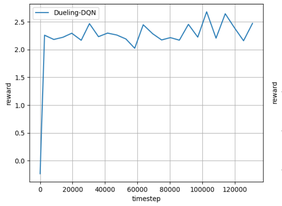
Results and Discussion

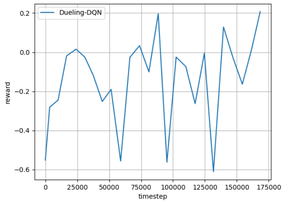
When training against random agent, all four models show significant improvement of over 2 big blind per hand. When training against rule-based agent, only two models show small improvement. Meanwhile, two variants that we implemented, Dueling DQN and Noisy DQN, further improved the performance of DQN.

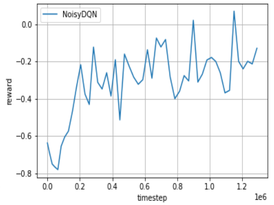
We also highlight that rule-based agent can only beat random agent by a small margin of around 0.7 BBPH. This reflects the mixing strategy of the rule-based agent, which makes it very hard to be beaten on a consistent basis. We think that this mixing strategy is a realistic reflection of professional human poker players, who try their best not be exploited by others in a zero-sum game of Texas Hold'em. Indeed, this is what we found in the reward over time during training. Reward of all DQN agents fluctuate, even in the last few epoch of training against rule-based agent.

When we train the Dueling DQN with a random agent, we can see that the average reward it gets is slightly higher than the reward that the double DQN gets when it was trained with the same random agent. One of the improvements we observed is that the Dueling DQN takes less training time compared to the Double DQN to get the similar rewards.

	Performance Improvement (big blinds per hand)				
trained with	Double DQN (DDQN)	Dueling DQN	Noisy DQN	Neural Fictitious Self-Play	Rule-based Agent
Random Agent	2.2479	2.3497	2.3908	2.4295	0.6945
Rule-based Agent	-0.1262	0.0059	0.0705	-0.1285	NA







References

Fortunato, M., & Azar, M.G., Piot B., Menick J., Osband I., Graves A., Mnih V., Munos R., Hassabis D., Pietquin O., Blundell C. & Legg S. (2017) Noisy Networks for Exploration. In ICLR, 2018.

Heinrich, J & Silver, D (2016) Deep Reinforcement Learning from Self-Play in Imperfect-Information Games. arXiv:1603.01121

Hasselt H. V., Guez A. & Silver D. (2015) Deep Reinforcement Learning with Double Q-learning. In AAAI, 2016.

Schaul T., Quan J., Antonoglou I. & Silver D. (2015) Prioritized Experience Replay. In ICLR, 2016.

Wang Z., Schaul T., Hessel M., Hasselt H. V., Lanctot M. & Freitas N. D. (2015) Dueling Network Architectures for Deep Reinforcement Learning. arXiv:1511.06581

Zha D. et al., (2020) RLCard: A Platform for Reinforcement Learning in Card Games. In IJCAI, 2020.