Building Poker Agent Using Reinforcement Learning with Neural Networks

Annija Rupeneite
Faculty of Computing, University of Latvia, 19 Raina blvd., LV-1586 Riga, Latvia

Keywords: Poker Game, Reinforcement Learning, Neural Networks.

Abstract: Poker is a game with incomplete and imperfect information. The ability to estimate opponent and interpret its actions makes a player as a world class player. Finding optimal game strategy is not enough to win poker game. As in real life as in online poker game, the most time of it consists of opponent analysis. This paper illustrates a development of poker agent using reinforcement learning with neural networks.

1 STAGE OF THE RESEARCH

Poker is a game with incomplete and imperfect information. The ability to estimate opponent and interpret its actions makes a player as a world class player. Finding optimal game strategy is not enough to win poker game. As in real life as in online poker game, the most time of it consists of opponent analysis.

Author is working on development of poker agent that would find optimal game strategy using reinforcement learning (RL) in combination with artificial neural network (ANN) for value function approximation.

2 OUTLINE OF OBJECTIVES

This paper illustrates a development of poker agent using reinforcement learning with neural networks. Complete poker agent should have an ability to create optimal game strategy that makes decisions based on information:

- Hand strength/potential estimation;
- Table card estimation;
- Opponent hand strength prediction;
- Opponent classification (tight/loose - passive/ aggressive);
- Current state evaluation using neural network.

AI Poker agent should be able to find an optimal strategy by itself (unsupervised learning) using information given above. It also should be able to adapt opponent play style and change its strategy during the game.

3 RESEARCH PROBLEM

Games are to AI as grand prix racing is to automobile design.

Poker game has become a field of interest for artificial intelligence technologies. There are some concealed cards in the game that makes impossible to calculate the final outcome of the hand. Therefore, artificial intelligence approach is used more often to develop online poker agents.

Poker game can be defined as a partially observable Markov decision process (POMDP). There is no complete and ready solution for POMDP. Reinforcement learning technologies allow to create an agent for Markov decision process (MDP), but it can’t make model for POMDP - it is impossible to define current state and calculate value function in uncertain environment. To solve POMDP with reinforcement learning neural network is used for value function approximation. Value function approximation with neural network allows to estimate a current state in uncertain environment.

Most of the known research work on poker game with AI includes opponent model with neural networks (Davidson, 1999) or reinforcement learning for finding optimal game strategy (Teófilo, Reis, Cardoso, Félix, Sêca, Ferreira, Mendes,Cruz, Pereira, Passos, 2012). However none of them uses both these technologies together. Reinforcement learning together with neural networks have successfully been applied for different control problems - control of gas turbine (Schafer, 2008) and motor-control task (Coulom, 2002).

Given research describes poker agent development using RL with ANN.

4 STATE OF THE ART

Texas Hold'em is one of the most popular forms of poker. It is also a very complex game. There are several factors that make poker game as uncertain environment (concealed cards, bluffing). These characteristics make poker partially observable Markov Decision process that has no ready solution.

Reinforcement learning together with neural network for value function approximation provides a solution for uncertain environment agent. This paper gives a brief description of information needed to develop agent poker game:

- Poker game rules;
- Definition of the partially observable Markov decision process;
- Neural network theory;
- Reinforcement learning theory.

4.1 Poker Game

Poker is a game of imperfect information in which players have only partial knowledge about the current state of the game (Johanson, 2007). Poker involves betting and individual play, and the winner is determined by the rank and combination of cards. Poker has many variations - in experiments, and data analyses author uses Texas hold'em poker game version. Texas hold'em consists of two cards dealt to player and five table cards. Texas hold'em is an extremely complicated form of poker. This is because the exact manner in which a hand should be played is often debatable. It is not uncommon to hear two expert players argue the pros and cons of a certain strategy (Sklansky, Malmuth, 1999).

Poker game consists of 4 phases - pre-flop, flop, turn, river. On the first phase (pre-flop) two cards are dealt for every player. On the second phase (flop) three table cards are shown. On next phase (turn) fourth card is shown and finally on the last phase (river) table fifth card is shown and winner is determined. Game winner is a player with the strongest five card combination. Possible card combinations are (starting from the highest rank) Straight flush, Royal flush, Four of a kind, Full house, Flush, Straight, Three of a kind, Two pair, One pair, High card.

4.2 Partially Observable Markov Decision Process

Markov decision process can be described as a tuple (S, A, P, R), where

- S, a set of states of the world;
- A, a set of actions;
- P:S×S ×A →[0,1], which specifies the dynamics. This is written P(s'|s,a), where

∀s ∈S ∀a ∈A ∑s'∈S P(s'|s,a) = 1.

In particular, P(s'|s,a) specifies the probability of transitioning to state s' given that the agent is in a state s and does action a.

- R:S×A ×S →R, where R(s,a,s') gives the expected immediate reward from doing action a and transitioning to state s' from state s (Poole and Mackworth, 2010).

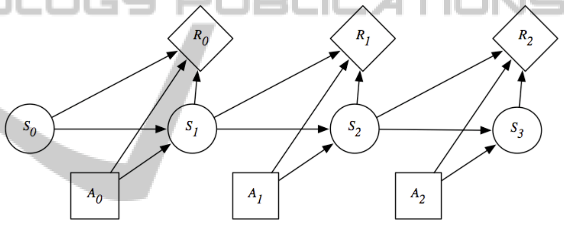


Figure 1: Decision network representing a finite part of an MDP (Poole and Mackworth, 2010).

Partially observable Markov decision process is a formalism for representing decision problems for agents that must act under uncertainty (Sandberg, Lo, Fancourt, Principe, Katagiri, Haykin, 2001).

POMDP can be formally described as a tuple (S, A, T, R, O, Ω), where

- S - finite set of states of the environment;
- A - finite set of actions;
- T: S × A → Δ(S) - state-transition function, giving a distribution over states of the environment, given a starting state and an action performed by the agent;
- R: S × A → R - the reward function, giving a real-values expected immediate reward, given a starting state and an action performed by the agent;
- Ω - finite set of observations the agent can experience;
- O: S × A → Δ(Ω) - the observation function, giving a distribution over possible observations, given a starting state and an action performed by the agent.

Poker game can be described as POMDP - there are concealed cards during the game what makes it partially observable. This characteristic makes poker game as testbed for AI research - there is no mathematical model for optimal game strategy.

4.3 Neural Networks

A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

1. Knowledge is acquired by the network through a learning process.

2. Interconnection strengths known as synaptic weights are used to store the knowledge (Haykin , 1999).

Basically, learning is a process by which the free parameters (i.e., synaptic weights and bias levels) of a neural network are adapted through a continuing process of stimulation by the environment in which the network is embedded (Sandberg, Lo, Fancourt, Principe, Katagiri, Haykin, 2001).

Neural network has interconnections between neurons in different layers. The first layer has input neurons, which send data via synapses to the next layer of neurons and so on till the last layer of neurons. Neural network is described by parameters:

- Interconnection pattern between different layers of neurons;
- Learning process for updating weights;
- Activation function that calculates neuron output from input with weights.

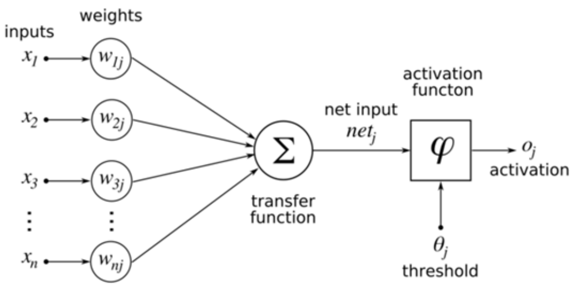


Figure 2: Basic structure of an Artificial Neural Network (ANN).

In the given research neural network technology is used for several purposes:

- To model opponent based on previous actions;
- For value function approximation with reinforcement learning.

4.4 Reinforcement Learning

Reinforcement learning (RL) is learning what to do - how to map situations to actions - so as to maximize a numerical reward signal. The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. (Sutton & Barto , 1998).

These characteristics is the most important reason why RL is used to make different game programs. For example, in 1995, G.Tesauro created TD-Gammon program for game of Backgammon, where innovation was in the method how it learned its evaluation function (Tesauro, 1995).

RL problem can be formalized as follows. The environment is modeled as a stochastic finite state machine with inputs (actions sent from the agent) and outputs (observations and rewards sent to the agent):

- State transition function P(X(t)|X(t-1),A(t));
- Observation (output) function P(Y(t) | X(t), A(t));
- Reward function E(R(t) | X(t), A(t)).

The agent's goal is to find a policy and state-update function so as to maximize the expected sum of discounted rewards (Murphy, 1998).

There are several RL algorithms to solve problems described as MDP. Learning approaches can be classified as indirect learning and direct learning. Direct learning includes a value-function based learning - temporal difference (TD) learning, Q-learning. In this research, author focuses on value-based methods.

Value Functions are state -action pair functions that estimate how good a particular action will be in a given state, or what the return for that action is expected to be.

The value of taking action a in the state s under a policy π is called Q-value denoted as Qt(st,at).

Reinforcement learning model consists of:

1. a set of possible states, represented by S;
2. a set of actions, A;
3. a set of numerical rewards R (Patel and Barve, 2014).

In Q-learning and related algorithms, an agent tries to learn the optimal policy from its history of interaction with the environment. Q*(s,a), where a is an action and s is a state, is the expected value (cumulative discounted reward) of doing a in the state s and then following the optimal policy.

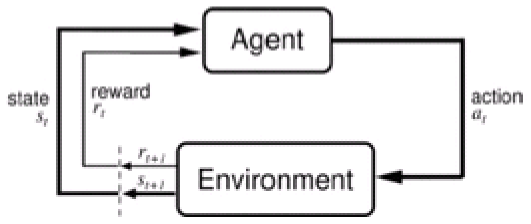


Figure 3: Standard Reinforcement Learning Model. (Patel and Barve, 2014).

Q-learning uses temporal differences to estimate the value of Q*(s,a). In Q-learning, the agent maintains a table of Q[S,A], where S is the set of states, and A is the set of actions. Q[s,a] represents its current estimate of Q*(s,a) (Poole and Mackworth, 2010).

Reinforcement learning technologies allow Markov decision process agent development, but standard RL algorithm can't solve POMDP because of uncertainty and lack of information about the current state.

4.5 Related Work

There have been many different approaches in online poker agent development. They can be classified as AI based or based on mathematics and statistics. University of Alberta Computer Poker Research Group has developed several online poker agents that use game theory approach (Nash equilibrium), statistics and also there are some modification's using artificial neural networks: Loki (1997), Poki - improved Loki (1999) PsOpti/Sparbot (2002), Vexbot (2003), Hyperborean (2006), Polaris (2007), Hyperborean (2007), Hyperborean Ring (2009).

There are also several research works that use only AI technologies. For example, Néill Sweeney, David Sinclair in their work "Applying Reinforcement Learning to Poker" describes the application of basic reinforcement learning techniques to the game of multi-player limit hold'em poker. A.Davidson, D.Billings, J.Schaeer, D.Szafron in their work "Improved Opponent Modeling in Poker" reports progress achieved by improved statistical methods, which were suggested by experiments using artificial neural networks.

None of these experiments have used reinforcement learning together with artificial neural networks to develop poker agent.

There are some research works using reinforcement learning with neural network for value function approximation in other areas. Anton Maximilian Schafer in his research work "Reinforcement Learning with Recurrent Neural Networks" has shown successful application of these technologies for controlling a high-dimensional dynamic system with continuous state and action spaces in partially unknown environment like a gas turbine.

5 METHODOLOGY

Given research describes poker agent developed using several AI technologies - reinforcement

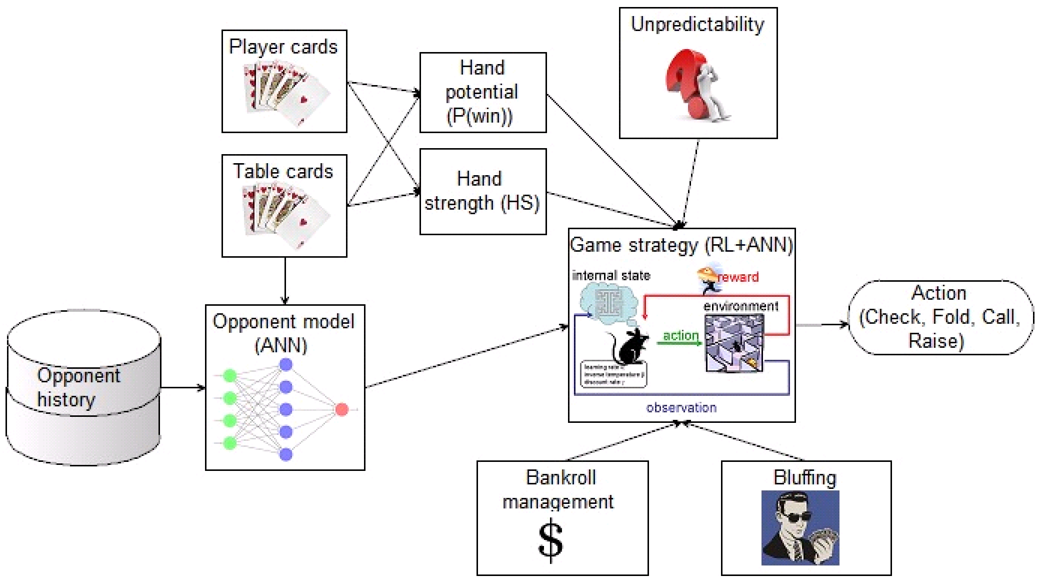


Figure 4: Poker agent architecture.

learning and neural networks. To be a world class poker player it is not enough to optimize only one part of game. Good poker agent should be able to process different type information to make a decision about action. This information includes:

- Hand potential evaluation;
- Hand strength evaluation;
- Bankroll management;
- Opponent modeling;
- Unpredictability;
- Bluffing.

Based on information given above poker agent should choose action - check, fold, raise or call. Full poker agent model is shown in Figure 4. Author examines first four information types in this research.

5.1 Hand Strength and Potential

The hand strength (HS) is the probability that a given hand is better than that of an active opponent.(Felix,Reis, 2008)

To quantify the strength of a poker hand where its result expresses the strength of a particular hand in percentile (i.e. ranging from 0 to 1), compared to all other possible hands Effective Hand Strength (EHS) algorithm is used. Algorithm conceived by computer scientists Darse Billings, Denis Papp, Jonathan Schaeffer and Duane Szafron.

Hand potential is calculated:

$$P(win) = HS \times (1 - NPot) + (1-HS) \times PPot. \qquad (1)$$

Where

- P(win) - probability of winning at the showdown;
- HS - current Hand Strength (not taking into account potential to improve or deteriorate, depending on upcoming table cards);
- NPot - negative potential (probability that current hand, if the strongest, deteriorates and becomes a losing hand);
- PPot - positive potential (probability that current hand, if losing, improves and becomes the winning hand) (Felix,Reis, 2008).

5.2 Opponents' Modeling

There are many research work done which prove that opponent exploration is one of the key factors for a good poker player (Felix, Reis, 2008).

In the given research, 2 approaches are combined for opponent modeling - opponent classification and

for opponent modelling - opponent classification and

opponent's hand strength evaluation. Opponent classification by play style is quite simple - it can be calculated by a formula, but to predict opponent's hand strength neural network technology is used. Neural networks allow to process different input data to get approximate assessment of hand strength.

5.2.1 Opponent Classification

Poker player can be classified under four categories of playing styles. Each style describes the frequency of play and how the player bets. Playing styles are loose/passive, tight/passive, loose/aggressive and tight/aggressive.

Loose/Tight Play Style

A tight player plays few hands and often folds. These players limit their play to only the best starting hands. Loose players play with a wide range of hands - they play many hands and are at the center of the action.

Player can be classified by loose/tight by percent of games they have played:

- Tight - plays <28% hands;
- Loose - plays >=28% hands.

Aggressive/Passive Play Style

Aggressive players bet and raise a lot and almost never check or call, but passive players usually check and call - they follow the action rather than take the lead. Aggressive players can win hands, even if they don't have the best cards.

Player can be classified as Aggressive or Passive by aggression factor (AF) (Li,2013). AF can be calculated with the formula:

$$AF = NumRaises/NumCalls. \qquad (2)$$

Where

- NumRaises – number of hands raised; NumCalls – number of hands called.

Player is classified as aggressive if AF >1 and passive if AF <=1.

Table 1: Player classification by play style (Felix, Reis, 2008).

	AF<=1	**AF>1**
%GP>=28	Loose Passive	Loose Aggressive
%GP<28	Tight Passive	Tight Aggressive

Figure 5 shows player classification and target play zone for the agent.

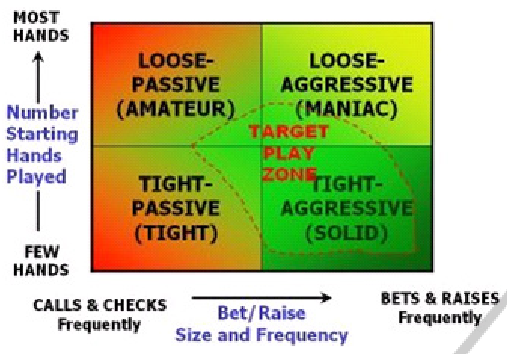


Figure5: The hand strength (HS) is the probability that a given hand is better than that of an active opponent.(Felix, Reis, 2008).

5.2.2 Opponent Modeling using Neural Networks

In the research, neural network - AI technology is used to predict opponent hand strength. Neural network has the ability to generalize data - a trained network can classify data from the same class as the learning data. Neural networks process information in a similar way the human brain does. The network is composed of a large number of highly interconnected processing elements (neurons) working in parallel to solve a specific problem. Neural networks learn by example (Stergiou, Siganos, 1995).

Multilayer feed-forward neural network architecture with supervised learning was used for the poker opponent model. Real online game poker data was used for experiments. Data consisted of more than 700 000 Texas hold'em poker game hands in a raw text format. Neural network system was created to predict opponent's hand strength, which would help a player to make a decision.

Data pre-processing is very important to achieve good results from machine learning technologies, including neural network.

To get structural and readable data, text files were processed and imported into MS SQL database. Data contained:

- Player action in every poker phase;
- Table cards;
- Table pot size;
- Every player pot size;
- Winner player;
- Player identifier;
- Cards for players, who made a showdown.

Given data was processed and transferred into a table that was used as input data for neural network:

- Hand and player identifier;
- Hand date and time;
- Player and table chip count;
- Player action done;
- Flag if player won hand;
- Flag if hand date is on weekend;
- Flag if only 2 players are left on river phase;
- Flag if player had won in last 10 games;
- Flag if player had ever bluffed;
- Table cards and player card;
- Table cards strength evaluation;
- Player cards strength evaluation.

Processed data was split into 2 parts - training data (80%), validation data and test data (20%). To analyze input data impact on results several combinations of input data was used.

Good poker opponent model is a part of good poker agent. Given results shows that neural network can predict poker opponent card strength in 78% of hands. Given neural network can be used to help with making the decision in poker game. Given experiments shows that input data have a huge impact on network result, and it can be improved by adding more pre-processed input data.

5.3 Game Strategy

Good poker game strategy should consider information above for making decision how to act (check, call, raise and fold). Algorithm for complex game such as poker should be able to estimate whole game result and possible total reward (not optimize one game phase).

Reinforcement learning provides technology to develop such model if game can be described as the Markov decision process.

Reinforcement learning can't be directly applied to poker game because of uncertainty. Poker game state can't be calculated, and we can't define value function. Instead of formula for action function approximation can be used. We can make value function approximation in poker game based on information available. Such information is: number of players, place on the table, table cards, hand strength, hand potential, opponent classification and opponent's hand strength evaluation. To get such value approximation we need technology that can form function with noisy, multidimensional data. Artificial neural network is such technology.

Aim of this research is to develop a model that can find optimal game strategy using reinforcement learning and neural network for value function using input data described above. Author has developed simple RL model with NN value function that can find optimal game strategy in 2 player poker game.

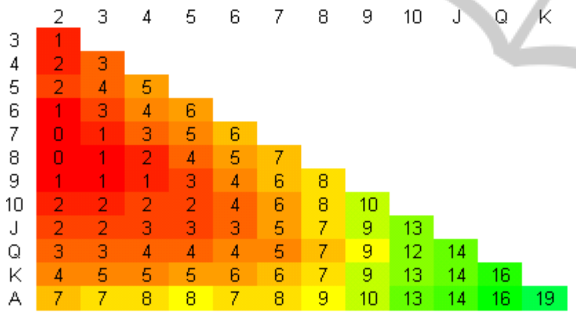
5.4 Agent Evaluation

To evaluate developed poker game agent various approaches will be used. AI poker game play results will be compared to 3 different poker agents that are developed during the research:

1) Random poker player;
2) Safe-game poker player;
3) Game strategy obtained from poker professionals.

Random poker player is a computer program that chooses action with random number generator.

Safe-game poker player is a strategy obtained from several poker rooms and poker tutorials for beginners. This approach is based on a theory of Probability - all 2 card hands are rated on a 0 to 40 scale.

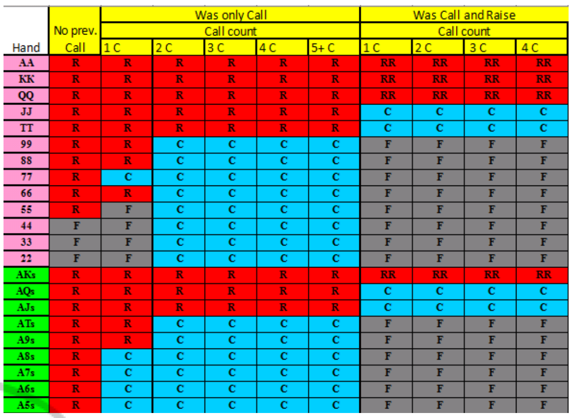


	2	3	4	5	6	7	8	9	10	J	Q	K
3	1											
4	2	3										
5	2	4	5									
6	1	3	4	6								
7	0	1	3	5	6							
8	0	1	2	4	5	7						
9	1	1	1	3	4	6	8					
10	2	2	2	2	4	6	8	10				
J	2	2	3	3	3	5	7	9	13			
Q	3	3	4	4	4	5	7	9	12	14		
K	4	5	5	5	6	6	7	9	13	14	16	
A	7	7	8	8	7	8	9	10	13	14	16	19

Figure 6: Unsuited Cards Power Rating (http://wizardofodds.com).

For example, Figure 6 shows power rating for each initial different suit 2-card hand if cards are from different suits. This strategy says, that player should play if hand rate is at least 13 or play with rate 19 in early position, but 10 in late position.

Game strategy obtained from poker professionals is similar to the Safe-game poker player, but it has a more detailed description for each game phase - action depends on player's position on the table and previous opponent's actions. This strategy is obtained by combining several poker professional strategies (Hilger, 2003 and Sklansky, 2005). Figure 7 shows middle position player strategy. For example, table shows that player with hand of the same suit ace and king should raise if there is no previous call or raise or there has been only raise, but it should re-raise if other players have raised and called.



	No prev.	Was only Call					Was Call and Raise			
		Call count					Call count			
Hand	Call	1 C	2 C	3 C	4 C	5+ C	1 C	2 C	3 C	4 C
AA	R	R	R	R	R	R	RR	RR	RR	RR
KK	R	R	R	R	R	R	RR	RR	RR	RR
QQ	R	R	R	R	R	R	RR	RR	RR	RR
JJ	R	R	R	R	R	R	C	C	C	C
TT	R	R	R	R	R	R	C	C	C	C
99	R	R	C	C	C	C	F	F	F	F
88	R	R	C	C	C	C	F	F	F	F
77	R	C	C	C	C	C	F	F	F	F
66	R	R	C	C	C	C	F	F	F	F
55	R	F	C	C	C	C	F	F	F	F
44	F	F	C	C	C	C	F	F	F	F
33	F	F	C	C	C	C	F	F	F	F
22	F	F	C	C	C	C	F	F	F	F
AKs	R	R	R	R	R	R	RR	RR	RR	RR
AQs	R	R	R	R	R	R	C	C	C	C
AJs	R	R	R	R	R	R	C	C	C	C
ATs	R	R	C	C	C	C	F	F	F	F
A9s	R	R	C	C	C	C	F	F	F	F
A8s	R	C	C	C	C	C	F	F	F	F
A7s	R	C	C	C	C	C	F	F	F	F
A6s	R	C	C	C	C	C	F	F	F	F
A5s	R	C	C	C	C	C	F	F	F	F

Figure 7: Game strategy for middle position.

Author has developed 3 type computer poker players that allow to compare and evaluate AI poker agent results.

6 EXPECTED OUTCOME

Based on reinforcement learning in combination with neural network techniques for problem solving, author proposed online poker agent development. This approach allows to solve poker game strategy optimization problem which can be described as POMDP.

Complete poker agent should be able to process following information for optimal strategy finding with RL:

- Hand potential and strength;
- Opponent play style (classification);
- Opponent hand strength evaluation with ANN:
- Value function approximation with ANN.

The future work tends to improve the neural network opponent model to achieve more precise prediction results. Several neural networks with different input data will be tested to find out the best one for the RL poker agent. Additionally testing for the proposed approach in different scenarios and with different learning data will be done.

REFERENCES

Coulom M.R., 2002. Reinforcement Learning Using Neural Networks, with Applications to Motor Control. PhD thesis, Institut National Polytechnique de Grenoble.

Davidson A., 1999. *Using Artifical Neural Networks to Model Opponents in Texas Hold'em*. University of Alberta

Davidson A., Billings D., Jonathan Schaeer, Duane Szafron, 2002. *Improved Opponent Modeling in Poker*. Artificial Intelligence - Chips challenging champions: games, computers and Artificial Intelligence Volume 134 Issue 1-2.

Felix D., Reis L.P., 2008. *An Experimental Approach to Online Opponent, Modeling in Texas Hold'em Poker*. Advances in Artificial Intelligence –SBIA 2008.

Félix D. and Reis L.P., 2008. *Opponent Modelling in Texas Hold'em Poker asthe Key for Success*. ECAI 2008.

Haykin S.S., 1999. *Neural networks : a comprehensive foundation.* Upper Saddle River, NJ : Prentice Hall.

Hilger M., 2003, *Internet Texas Hold'em: Winning Strategies from an Internet Pro.* Dimat Enterprises, Inc.

Johanson M., 2007. *Robust strategies and counterstrategies: Building a champion level computer poker player.* In Masters Abstracts International, volume 46.

Li A., 2013. *Enhancing Poker Agents with Hand History Statistics*. Bachelor-Thesis, Technische Universitat Darmstadt.

Murphy K.P., 1998. *A brief introduction to reinforcement learning*. University of British Columbia.

Patel J.R. and Barve S.S., 2014. *Reinforcement Learning: Features and its applications*, International Journal of Computer Technology and Applications, volume 5 Issue 3.

Poole D. and Mackworth A., 2010. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press.

Sandberg I.W., Lo J.T., Fancourt C.L., Principe J.C., Katagiri S., Hayk S., 2001. *Nonlinear Dynamical Systems: Feedforward Neural Network Perspectives.* John Wiley & Sons,

Sklansky D, Malmuth M., 1999. *Hold'em Poker For Advanced Players.*Two Plus Two Pub.

Sklansky D., 2004. *The Theory of Poker*, Two Plus Two Publishing.

Stergiou C. and Siganos D., 1995. N*eural Networks*, Surprise 96 Volume 4 (Final Reports).

Sutton R.S. and Barto A.G., 1998. *Reinforcement Learning: An Introduction*. The MIT Press.

Szepesvari C., 2010. *Algorithms for Reinforcement Learning*. Morgan and Claypool Publishers.

Sweeney N., Sinclair D., 2012. *Applying Reinforcement Learning to Poker*.Compter Poker Symposium.

Teófilo L.F., Reis L.P., Cardoso H.L., Félix D., Sêca R., Ferreira J., Mendes P., Cruz N., Pereira V., Passos N., 2012. *Computer Poker Research at LIACC*. 2012 Computer Poker Symposium at AAAI.

Tesauro G., 1995. *Temporal Difference Learning and TD-Gammon*. Communications of the ACM, March 1995 / Vol. 38, No. 3.