Reinforcement Learning in the Game of Ludo

Jan-Ruben Schmid

University of Southern Denmark, Campusvej 55, 5230 Odense M, Denmark
jschm20@student.sdu.dk

Abstract. This paper demonstrates a method for an Artificial Intelligence (AI) Ludo playing agent. One method was implemented and compared against a method from a fellow student. The parameters of the method are determined by Q-Learning and multiple reward values. The final developed player has a win rate of 83% against one random player and 60% against three random players.

1 Introduction

Due to the increased amount of data and the variety of available data, machine learning becomes irreplaceable for more and more companies. It is already playing part in most industries, like financial services, health care, and transportation [1]. The Journal of Business Strategy announced last year, that "the wave of fully implementing AI is far from reaching is inflection point" [2]. Therefore, more skilled people are required who understand how the systems work to maintain and build them. This project describes an AI approach for the board game "Ludo". This is done by a method called 'Q-Learning", which is a reinforcement learning technique used for optimal learning in a Macrov decision process.
For this report the python library LudoPy is used with the rules described in the repository [3]. The implementation can be found at
https://github.com/janschmid/LudoPy.

2 Method

The AI player is realized with Q-learning, a model-free reinforcement learning technique [4]. By repeated retrying all actions in all states with a long-term discount reward, it learns which is the best overall action [5]. To apply Q-learning, the game has to be in state space representation.

2.1 State-space representation

The states of the game are represented in a discrete set of environment states and actions. The used states and actions are represented in table 1 and 2, where a brief description of each entry is given.

State	Description
Home	Piece is in home location
Safe	The piece can't get killed by an opponent, this is the case if it is "protected", "on a globe", or "in the goal zone"
Unsafe	The piece is not safe, but can't be reached by an opponent player within one dice roll
Danger	The piece not safe and within the range of one dice roll of the opponent

Table 1. States

Action	Reward	Description
Move out	0.25	Moving token out from Home position
Normal	0.01	Moving eyes of dice on board
Goal	0.8	Moving to goal position
Star	0.5	Move to star, this results in a "jump" to the next star
Globe	0.4	Move to a safe point
Protect	0.3	Move to a point where another piece as yourself is (can not get killed)
Kill	0.4	Kill an opponent piece
Die	0	Move to field where opponent is safe, e.g. protect or globe
Goal Zone	0.4	Move into goal zone
Nothing	0	No action is possible

Table 2. Actions

2.2 Q-Learning

The Q-Learning formula is given by:

$$\underbrace{\text{New}Q(s,a)}_{\text{New Q-value}} = Q(s,a) + \underbrace{\alpha}_{\text{Learning rate}} [\underbrace{R(s,a)}_{\text{Reward}} + \underbrace{\gamma}_{\text{Discount factor}} \overbrace{\max Q'(s',a')}^{\text{Maximum predicted reward, given new state and all possible actions}} - Q(s,a)] \quad (1)$$

where s represents the state and a the choosen action. The learning rate defines to what extent the new acquired information are taken into account. The factor is bounded between [0,1], where 1 makes the agent using only the newest information. 0 means that the agent learns nothing, since the "estmiate of optimal future value" is multiplied with the factor, see equation 1.
The discount factor determines the importance of future rewards. If the discount factor is 0, the maximum predicted reward in the next state is not taken into account, which results in a short-sighted agent.
While the model is trained, an ϵ-greedy exploration is used with a fixed rate. The value is a trade-off between exploitation and exploration, which is known as the tow-armed Bernoulli bandit problem [6]. The ϵ-greedy mechanism selects

a random action with the probability ϵ and the highest Q-value with the probability $1 - \epsilon$.
The maximum predicted reward is calculated by taking all possible actions from the piece at the new location into account, where the highest possible reward is used. For the reward values, the proposed values from the lecture were used, see table 2. Beyond the state space representation, the chosen values of the learning rate α and the discount factor λ have a major impact on the results, which will be described in following.

2.3 Choosing the best learning rate and discount factor

As mentioned in section 2, the learning rate and discount factor have a high impact on the overall performance. To find the best values, multiple games were played against one and three random players. Figure 1 visualize the overall change of the Q-Table between each played game. It is noted that only the change of value is relevant. It can be seen, that the model converges faster with a higher learning rate. The change of Q-values is approximately constant after 25 games with a learning rate of 0.4 and 200 games with a learning rate of 0.06, but the overall bias is lower with a slower learning rate. Due to the comparable small amount of games required until the Q-values are settled over the statistical nature of the game, a small learning rate of 0.6 is used in following.

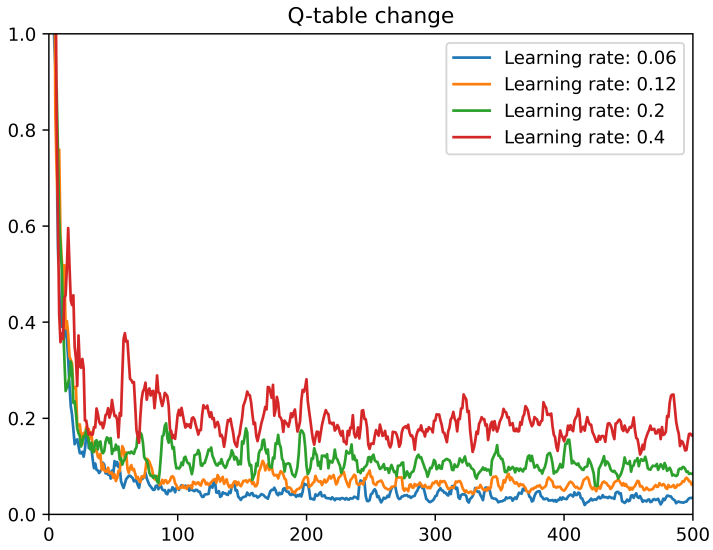

Fig. 1. Change of Q-values, constant discount factor

The Q-table is determined by playing 500 games with a $\epsilon - greedy$ value of 0.85, which means that in 15% of the cases a random action is chosen, in 85% of the cases the highest Q-value is chosen. The learning rate is set to 0.06. The performance of the difference in the discount factor is visualized in figure 2.

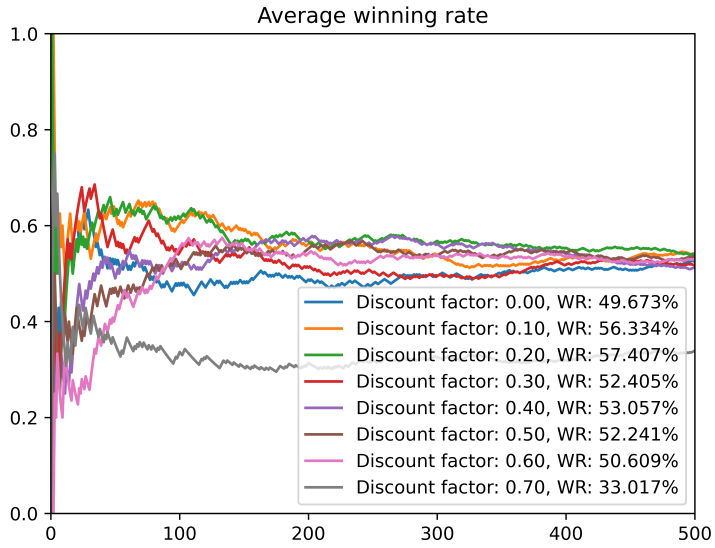

Fig. 2. Change of Q-values, fixed learning rate of 0.06

Figure 2 visualizes that the best performance is achieved by a discount factor of 0.2. While the change of the discount factor seems to be marginal in the range of [0,0.6], a discount factor higher than 0.6 results in a drastically decreasing performance. A possible explanation could be that the future award is the main bearing for the choice, while the agent is heading towards the next possible rewards value instead of the current one.

2.4 Improvement

The chosen action of the presented state space implementation takes the current state into account, as well as the action, but not the position of the next state. If two pieces have, for example an "unsafe" state and the only possible action for both pieces is "normal", no distinction can be made. If the movement of piece 1 results in the "unsafe" state, and the move of piece 2 in the "danger" state, the movement of piece 1 should be preferred. This behavior can be achieved by expanding the action table in an action-state based table, where the new table

has a size of *number of actions * number of states*.
The reward value is calculated by the addition of a state-dependent constant factor to the action. The state-based action reward values are listed in table 3. If the modified table is applied, a movement from the "normal" into an "unsafe"

State	State-based reward value
Home	+0
Safe	+0.2
Unsafe	+0.1
Danger	+0.0

Table 3. Modified reward table

state would have a reward value of *(0.01+0.1=0.11)*, into a "Danger" state *(0.01+0.0=0.01)*, where the first movement is the preferred one. It should be noted that multiple of the new state-based actions can not be played due to the logic of the game, but to the fact that it has no negative impact, it is chosen due to the ease of implementation, see section 7. If not mentioned differently, modified rewards table is used in following of the report, also called "extended state space".

2.5 Comparison of methods

This described method is compared with Nathan Durocher's approach [7], where a more aggressive playing style is approached by high rewards for moves like killing the opponent player. While the general approach is identical, the differences will be highlighted in following.

The state space representation has slight changes. Nathan Durocher's report highlights the improvement of the behavior in the goal zone. An additional state "Goal Zone" is added, whereby no distinction is made between the states "Danger" and "Unsafe". The additional action "Overshoot Goal" prevents the player to miss the goal and thereby waste time in the goal zone. The risk of killing the own piece by moving to an opponent's safe spot is not taken into account. [7] The reward table of Nathan Durocher's approach is visualized in table 4

Open	Normal	Goal	Star	Globe	Protect	Kill	Overshoot	Goal Zone
0.25	0.0001	0.9	0.5	0.4	0.2	0.5	0	0.4

Table 4. Action rewards

3 Results

In this section the results with the determined parameters, described in section 2.3, will be described. The parameters can be seen in table 5. A winning rate of

Learning rate	Discount factor
0.06	0.2

Table 5. Chosen parameters

83% against one random player and 60% against two random players could be achieved. The accumulated average winning rate is plotted in figure 3.

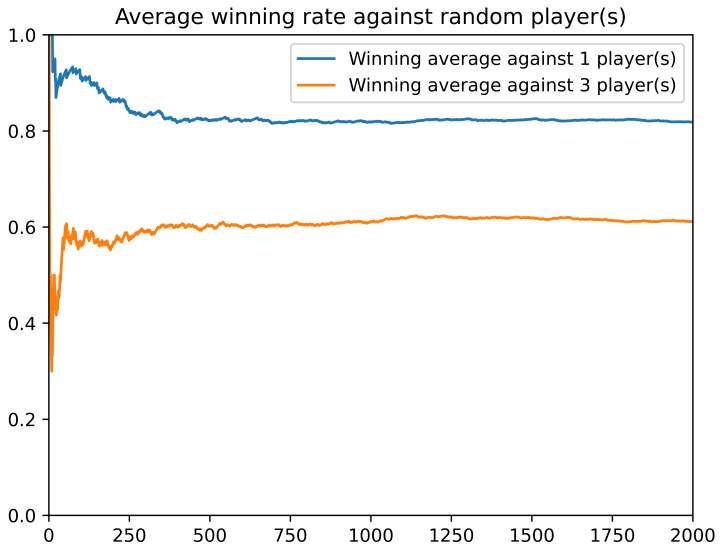

Fig. 3. 2000 games against random player(s)

Figure 5 visualizes a direct game against the approach from Nathan Durocher, where the aggressive playing style has an overall advantage. Figure 4 visualizes the result against the approach from Nathan Durocher, as well as two random players, where only the winning average of the non-random players are plotted. It can be seen that the choice of the previous described "balanced" approach has a clear advantage.

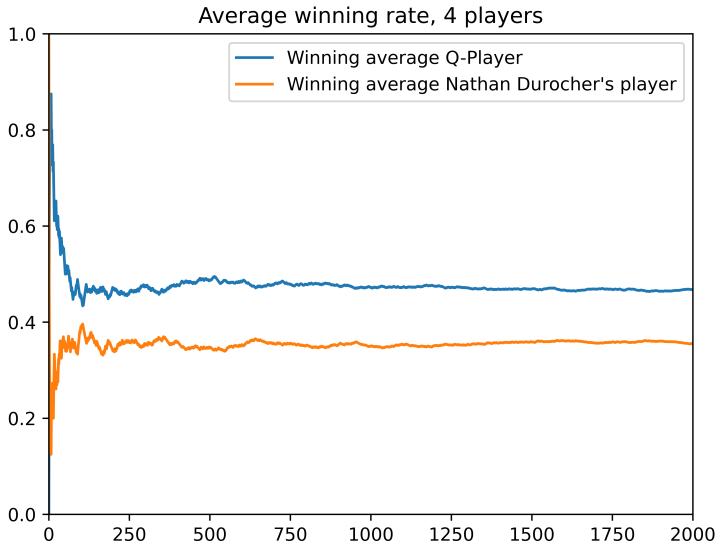

Fig. 4. 2000 games against two random players and Nathan Durocher's player

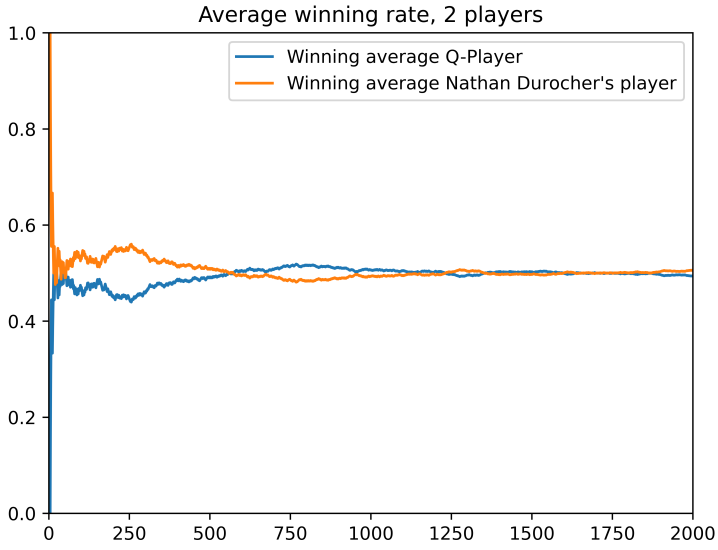

Fig. 5. 2000 games against Q-Player Nathan Durocher's player

The effect of the expanded state space is visualized figure 6, where the winning average of the extended state space and original state space is plotted. In both cases 2000 games against three random players are played. The extended state space achieves slightly better results.

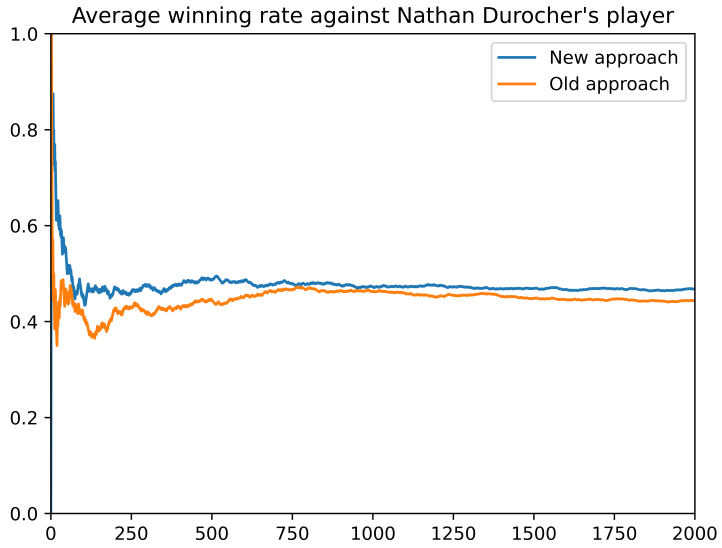

Fig. 6. 2000 games against Nathan Durocher's player, original (Old approach) and extended state space (New approach)

4 Analysis & Discussion

It can be seen that the selection of the discount factor has a significant effect on the overall result. While a higher learning rate accelerates the training of the model, the achieved values have a higher bias and the performance is therefore better with a lower learning rate. The balanced approach with a precise differentiation of the hazard potential has a comparable high performance against three opponent players, while the full potential is not played out in a direct match. Even the fact that not all actions are possible in the expanded state space, the approach returns better results.
There are two effects which lead to the relatively better results against three opponent players. The first is that high kill rewards seem to return the best results while playing against one player. A more conservative approach where a safe spot gets a high reward seems to work better against more players. The second

effect could be a precise differentiation of the hazard potential. If more opponent pieces are on the board, it is critical to keep the pieces as safe as possible and be capable to have an advanced threat detection.

5 Conclusion

The implemented Q-learning achieves significant higher results than a random behavior. The balanced approach returns the best results against multiple opponents, where the improved method of the state-action based action table contains more differentiated states, which result in an overall better performance. The agent was trained using unsupervised learning against random players. The determined values were used against another intelligent Q-learning approach and lead to satisfying results. Furthermore, the report highlights the importance of tuning values like learning rate and discount factor which were determined by a manual approach. More accurate values could be achieved by an iterative algorithm.
No attention was given to the reward values, where the significance could be seen in the comparison to the approach from a fellow student. Further work could be done in the determination of optimal reward values which could be achieved by a generic algorithm. Training against more intelligent player could have also be beneficial which could be determined in future work.

6 Acknowledgments

The author thanks Nathan Durocher for the use of his Q-learning player, as well as the constant exchange during the project duration. A thank you goes also to Simon Soerensen who implemented and maintains ludo for python, which saved many hours of development. Furthermore the autor would like to thank Poramate Manoonpong, for the project proposal and transferred knowledge throughout the semester.

References

1. Machine Learning `https://www.sas.com/sk_sk/insights/analytics/machine-learning.html`: (accessed 22 May 2020)
2. Lichtenthaler, U. (2020). Beyond artificial intelligence: why companies need to go the extra step. Journal of Business Strategy.
3. LudoPy `https://pypi.org/project/ludopy/`
4. Watkins, Christopher JCH, and Peter Dayan. "Q-learning." Machine learning 8.3-4 (1992): 279-292.
5. Watkins, Christopher John Cornish Hellaby. "Learning from delayed rewards." (1989).
6. Granmo, Ole-Christoffer. "Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton." International Journal of Intelligent Computing and Cybernetics (2010).
7. Nathan Durocher Ludo_Qlearning `https://github.com/Natenumber12/LUDO_QLearning` (accessed 23 May 2021)

7 Appendix

	Move out	Normal	Goal	Star	Globe	Protect	Kill	Die	Goal Zone	Nothing
Home	0.587458	0	0	0	0	0	0.710216	0	0	0
Safe	0	0	0	0	0	0	0	0	0	0
Unsafe	0	0	0	0	0	0	0	0	0	0
Danger	0	0	0	0	0	0	0	0	0	0

Table 6. Q-Table, state: Home

	Move out	Normal	Goal	Star	Globe	Protect	Kill	Die	Goal Zone	Nothing
Home	0	0	0	0	0	0	0	0	0	0
Safe	0	0	1	0.05069	0.192206	0.101458	0	0.033883	0.799954	0.190581
Unsafe	0	0	0	0.050657	0.723512	0.606011	0	0.120942	0.791798	0
Danger	0	0	0	0.092691	0.729948	0.038294	0	0.033883	0.787674	0

Table 7. Q-Table, state: Safe

	Move out	Normal	Goal	Star	Globe	Protect	Kill	Die	Goal Zone	Nothing
Home	0	0	0	0	0	0	0	0	0	0
Safe	0	0.252331	0	0.693473	0	0	0.642327	0.006	0	0
Unsafe	0	0.253734	0	0.696193	0	0	0.641092	0.030949	0	0
Danger	0	0.253828	0	0.700674	0	0	0.631659	0.018377	0	0

Table 8. Q-Table, state: Unsafe

	Move out	Normal	Goal	Star	Globe	Protect	Kill	Die	Goal Zone	Nothing
Home	0	0	0	0	0	0	0	0	0	0
Safe	0	0.144339	0	0.607438	0	0	0.091744	0	0	0
Unsafe	0	0.142031	0	0.60728	0	0	0.517323	0	0	0
Danger	0	0.150199	0	0.602768	0	0	0.522481	0.005203	0	0

Table 9. Q-Table, state: Danger