Design of an Artificial Game Entertainer by Reinforcement Learning

Takanobu Yaguchi
Kyoto Institute of Technology
Kyoto, Japan
yaguch8t@cis.is.kit.ac.jp

Hitoshi Iima
Kyoto Institute of Technology
Kyoto, Japan
iima@kit.ac.jp

***Abstract*—Games are often used for performance evaluation of artificial intelligence (AI) methods. Most AI studies using the games aim to design a computer player which plays a game better than humans. On the other hand, video game companies develop one-to-one games such as Go and Reversi and design computer opponents which entertain human players to have them play the games a lot. However, it takes much time to design such computer opponents. In this paper, we propose a reinforcement learning method for automatically designing an AI player entertaining the human players, especially those who are not good at playing games, in the one-to-one games. There are several ways to entertain them. One of the ways is to use a computer opponent which is neither too strong nor too weak, and the proposed method designs such an artificial game entertainer. The performance of the proposed method is evaluated through numerical experiments.**

***Index Terms*—Reinforcement learning, artificial intelligence, machine learning, game**

I. Introduction

In recent years, studies on artificial intelligence (AI) have attracted attention. The performance of AI methods is often evaluated by games [1]–[7]. Most AI studies using the games aim to design a computer player which plays a game better than humans, and AlphaGoZero [8] is one of the most successful studies. However, such AI players are too strong to entertain human players. Video game companies develop one-to-one games such as Go and Reversi and design computer opponents entertaining the human players to have them play the games a lot. Nevertheless, it takes much time to do it [9]. It is important to develop methods for automatically designing an artificial game entertainer; however there are few such studies.

In this paper, we propose a reinforcement learning method for automatically designing an AI player entertaining the human players, especially those who are not good at playing games, in the one-to-one games. The human players have their respective preferences, and therefore there are several ways to entertain them [10]. One of the ways is to use a weak AI player and to make them win overwhelmingly. Another is to use an AI player which is neither too strong nor too weak because humans can be satisfied when they achieve a moderately hard task [11]. The proposed method designs the latter artificial game entertainer, and we propose a reward setting useful for doing it.

978-1-7281-4533-4/20/$31.00 ©2020 IEEE

The AI player designed by the proposed method learns through playing against a computer opponent instead of a human player. Although the computer opponent used in AlphaGoZero is the AI player itself, such an opponent is inappropriate in the proposed method because the purpose of the AI player is different from that of the human player. Instead, we prepare a more appropriate opponent. The performance of the proposed method is evaluated through numerical experiments, where Reversi is used as a case study.

II. Learning Problem

This section defines our reinforcement learning problem in one-to-one games. In a game, two players take their actions in their respective turns. Depending on the actions, they get their respective scores. The player having more score is a winner when the game ends. The game is deterministic and has no random factors. The players can gain all information on the game. The one-to-one games described in the above include many traditional board games such as Go and Reversi. Such games have been developed as also video games in which a human plays against a computer player. In order to design the computer player, its strategy, that is, its actions must be determined. The objective of the proposed method is to design an AI player which is neither too strong nor too weak, as mentioned in the previous section. From this aspect, our reinforcement learning problem is defined as finding the actions of the AI player getting its score which is equal or close to the human's one.

III. Proposed Method

In this section, we propose a reinforcement learning method for finding appropriate actions of the AI player for the problem defined in the previous section. A popular reinforcement learning method is Q-learning [12], which uses state-action values. However, the numbers of states and actions tend to be enormous in a game, and it may take much time to learn the appropriate actions. To learn them rapidly, we adopt the temporal difference (TD) method [13] which uses state values. In the TD method, the AI player at a state s takes an action, and it gets the next state s' and a reward r. Then, the value $V(s)$ of the state is updated by

$$V(s) \leftarrow (1-\alpha)V(s) + \alpha(r + \gamma V(s')) \tag{1}$$

where α is the learning rate parameter, and γ is the discount rate parameter. The AI player plays a game against its opponent many times and learns based on the result of the play.

A. Reward Setting

The purpose of learning is that the AI player gets its score which is equal or close to the opponent's one, as described in Section II. A simple reward setting for realizing it is to give a reward if and only if the AI player gets the same score as the opponent at the end of a game. However, the game hardly ends with the same scores, which is likely to deteriorate the learning speed. Another probable reward setting is to give a reward if the difference in scores is smaller than a certain threshold. An appropriate threshold, however, must be given, and it takes time and effort to determine the threshold. To resolve these problems, we propose to give a larger reward for a smaller difference in the scores. Specifically, the reward *r* is defined as

$$r = \begin{cases} \frac{1}{|u_a - u_b|} & (u_a \neq u_b) \\ 1 & (u_a = u_b) \end{cases} \tag{2}$$

where u_a is the score of the AI player, and u_b is the score of the opponent. The reward is given only when the game ends. No reward is given when the game does not end. In the proposed reward setting, since a reward is given even if the game does not end with the same scores, the learning speed is accelerated. There exist no parameters, such as the threshold.

B. Opponent Setting

In applying a reinforcement learning method to a one-to-one game, a computer player must be prepared as the opponent of the AI player. In AlphaGoZero [8], the opponent is the AI player itself. The opponent setting, however, is inappropriate in the proposed method because the AI player's purpose is different from that of a human player, who is an opponent of the AI player after learning. Whereas the AI player's purpose is to get its score which is equal or close to the opponent, the human's purpose is to win the game. Therefore, the AI player fails to learn its superior actions if the opponent is the AI player itself. In order to learn them successfully, a more appropriate computer player should be set as the opponent. As mentioned in Section I, this paper focuses on designing the AI player for human players who are not good at playing games. They sometimes take good actions and sometimes bad ones, and their actions probably differ from each other. Such actions are similar to random ones. The random actions include both good and bad ones, and randomness brings different actions in every game. Thus, the proposed method uses a computer player which selects its action randomly. This computer player can be implemented easily and used for any one-to-one game.

C. Algorithm

The algorithm of the proposed method for one game is as follows.

1) Set the current state *s* as the initial state of the game.
2) The AI player decides its action at state *s* by the ϵ-greedy method. In the ϵ-greedy method, it chooses an action randomly with the probability ϵ, and with the probability $1 - \epsilon$ it chooses the action by which it reaches the next state whose value is the maximum.
3) The AI player gets the next state s'. If s' is the final state of the game, go to 5).
4) The AI player gets no reward ($r = 0$) and updates the state value *V*(*s*) by the equation (1). Set $s \leftarrow s'$ and go back to 2).
5) The AI player gets the reward *r* given by the equation (2) and updates the state value *V*(*s*) by the equation (1).

The AI player learns its actions by playing many games with the above algorithm.

IV. Experiments

Experiments are conducted by applying the proposed method to Reversi as a case study, and their results are shown to evaluate the effectiveness of the proposed method.

A. Reversi

Reversi or Othello is a board game for two players. There is a grid on the board, and the players take turns placing their respective disks into cells on the board. A player must place a disk such that some opponent's disks are between the placed disk and player's other disk. The opponent's disks are exchanged for player's ones. If the player cannot place a disk anywhere, the player must pass. When both the two players cannot place disks, the game ends. The score of each player is the number of the player's disks placed on the board, and the player having the majority of disks at the end of games is the winner.

When the normal board size is used, it takes much time to conduct all experiments because the state space is enormous. Therefore, our experiments use a 4×4 board whose size is reduced from the normal board size.

B. Experimental Method

The parameters of the proposed method are as follows.

- Number of games in learning: 40000
- Learning rate $\alpha = 0.3$
- Discount rate $\gamma = 0.999$
- Probability of randomly selecting an action $\epsilon = 0.2$

In order to verify the effectiveness of the reward setting proposed in Subsection III-A, the proposed AI player is compared with the following two AI players.

R1: gets the reward $r = 1$ only when a game is drawn.

R2: gets a reward when the difference in scores is smaller than a threshold *n* at the end of the game. The reward r is given by the following equation.

$$r = \begin{cases} \frac{1}{|u_a - u_b|} & (|u_a - u_b| \leq n \text{ and } u_a \neq u_b) \\ 1 & (u_a = u_b) \\ 0 & (\text{otherwise}) \end{cases} \tag{3}$$

Note that the parameter n must be given appropriately. In this experiment, we set $n = 4$.

In order to verify the effectiveness of the opponent setting proposed in Subsection III-B, the proposed AI player is compared with the following two AI players.

O1: plays against O1 itself.

O2: plays against a computer player which puts a disk at the position where the number of its disks is maximized.

O1 learns by using the experiences of the two players playing a game. Hence, the number of the experiences in O1 is twice as large as in the other AI players. O2 adopts a strategy which humans are likely to use because the purpose of Reversi is to get player's own more disks. Note that O2 cannot be applied to the other one-to-one games because it uses heuristics of Reversi. In contrast, the proposed AI player and O1 can be applied to any one-to-one game.

To evaluate the performance of each AI player, after learning, it plays games against 121 types of computer players instead of human players who are not good at playing games. The computer players are prepared in such a way that they moderately mimic the bad human players. As mentioned in Subsection III-B, the bad human players take an action which seems to be selected randomly. In Reversi, they are likely to use also the strategy of O2, as mentioned above. In addition, it is known that an easy-to-use and effective heuristic strategy is to put a disk at a corner of the board. From the above discussion, we prepare computer players using a combination of these three strategies. Specifically, they put their disk at a corner with the probability P_1 if they can. With the probability $1 - P_1$ they put it at a random position. If they cannot put it at any corner, they put it with the probability P_2 at the position where the number of their disks is maximized. With the probability $1-P_2$ they put it at a random position. (P_1, P_2) is set to $(0, 0), (0, 0.1), ..., (0, 1), (0.1, 0), ..., (0.1, 1), (0.2, 0)$, ..., $(1, 0.9)$, or $(1, 1)$, which means that 121 computer players are prepared.

In order to evaluate the performance of the whole proposed method, the proposed AI player is compared with the following computer player which does not use a machine learning method.

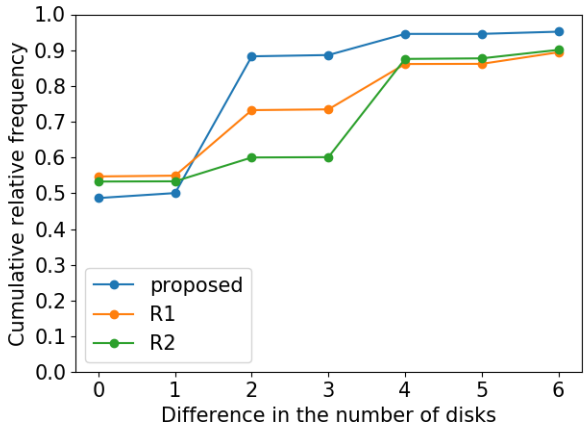
pr: puts its disk at the position where the difference D between the number of its disks and that of the opponent is minimized in the next its turn.

pr uses the strategy of the opponent in order to calculate the difference D. However, the strategy is actually unknown, and therefore pr is not useful practically. pr also plays games against the 121 computer players.

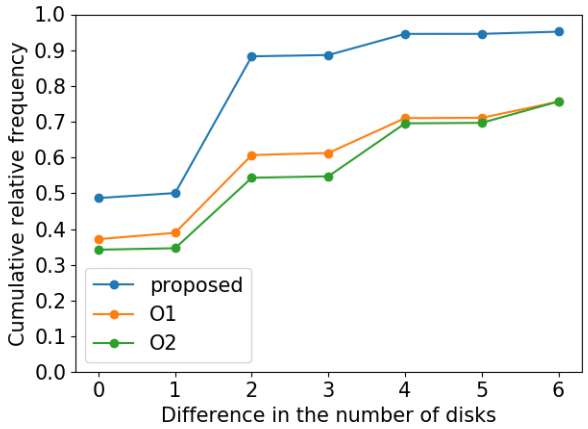
Each AI player alternates putting the first disk and putting the second disk. Since each learning method uses random numbers, it is performed 20 times to evaluate average performance.

C. Experimental Results and Discussion

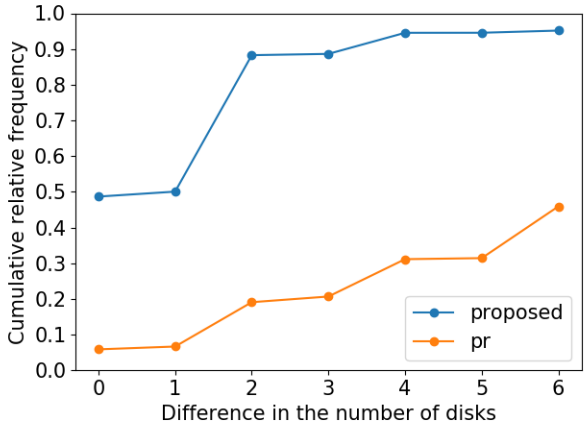
Figure 1 shows the results of games that various AI players and pr played against 121 computer players. Figure 1(a) shows comparison of the proposed AI player with R1 and R2. Figure 1(b) shows comparison with O1 and O2, and



(a) Comparison of the proposed AI player with R1 and R2



(b) Comparison of the proposed AI player with O1 and O2



(c) Comparison of the proposed AI player with pr

Fig. 1: Results of games which various AI players and pr played against 121 computer players

Fig. 1(c) shows comparison with pr. Each figure shows the cumulative relative frequency of differences in the number of disks. Therefore, if the cumulative relative frequency is larger in smaller differences in the number of disks, the fact means the AI player could learn better. Each line in each figure represents the average in twenty learning results. The experimental results show that the proposed method is superior to the other methods. The proposed AI player is tied with computer players in about half of games. In about 88 percent of the games, the difference in the number of disks is smaller than three. In about 95 percent of the games, it is smaller than five. Therefore, the proposed method can successfully design an AI player getting its score which is equal or close to an opponent's one.

If a human player notices intentional negligence of the proposed AI player, the human player is not entertained. Now we examine whether the proposed AI player takes such a negligent action or not. Because it is easy for human players to predict the outcome of a game in the endgame, they should notice the negligent action at that time. However, if the proposed AI player has only one possibly action, the action is not negligent. If it has more than one possibly action and takes a worse action, the action is negligent. In this paper, the worse action is defined as an action which results getting at least two disks fewer than the best action. From the above discussion, the rate *R* of taking the negligent action is defined as

$$R = \frac{N_w}{N} \tag{4}$$

where N_w is the number of times the proposed AI player takes worse actions, and N is the number of times it has more than one possibly action at the second and third turns from the last in games. Note that the last action is never negligent because the AI player cannot take any other action. For the proposed AI player, $R = 0.0464$ from experimental results. Therefore, the proposed AI player seldom takes the negligent action.

V. Conclusion

In this paper, we have proposed a reinforcement learning method for automatically designing an AI player which can entertain human players, especially those who are not good at playing games. There are several ways to entertain them. One of the ways is to use an AI player which is neither too strong nor too weak, and the proposed method designs the AI player. For doing so, we have particularly proposed a reward setting and an opponent setting. From the results of numerical experiments using Reversi, the AI player learned by the proposed method more successfully plays Reversi than by other methods, and therefore the proposed method is effective.

In the experiments, the performance of the proposed AI player is evaluated by making it play against 121 types of computer players instead of human players who are not good at playing games. However, it should be evaluated also through experiments in which the human players are the opponent of the proposed AI player. To conduct the experiments is future work. Although the TD method is used as the basic framework in this paper, it is expected that learning will become difficult if the number of states increases. In the future, the concept of the proposed method should be introduced into deep reinforcement learning, which is effective for such a large number of states.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP20K11988.

References

[1] P. Rohlshagen, J. Liu, D. Perez-Liebana, and S.M. Lucas, "Pac-Man Conquers Academia: Two Decades of Research Using a Classic Arcade Games," IEEE Transactions on Games, vol.10, No.3, pp. 253-256, 2018.

[2] N. Justesen, P. Bontrager, J. Togelius, and S. Risi, "Deep Learning for Video Game Playing," IEEE Transactions on Games, vol.12, No.1, pp. 1-20, 2020.

[3] J. Tsay, C. Chen, and J. Hsu, "Evolving Intelligent Mario Controller by Reinforcement Learning," Proceedings of 2011 Conference on Technologies and Applications of Artificial Intelligence, pp. 266-272, 2011.

[4] J. Quitério, R. Prada, and F. S. Melo, "A Reinforcement Learning Approach for the Circle Agent of Geometry Friends," Proceedings of IEEE Conference on Computational Intelligence and Games, pp. 423-430, 2015.

[5] V. Mnih, K. Kavukcuoglu, et al., "Human-level control through deep reinforcement learning," Nature, vol.518, pp.529-533, Feb. 2015.

[6] X. Guo, S. Singh, H. Lee, R.L. Lewis, and X. Wang, "Deep learning for real-time Atari game play using offline monte-carlo tree search planning," Advances in Neural Information Processing Systems, vol.4, pp.3338-3346, Jan. 2014.

[7] M. Abdollahi and C. Dadkhah, "Intelligent Android Game using Reinforcement Learning to Change the Enemy's Behavior," 2018 2nd National and 1st International Digital Games Research Conference: Trends, Technologies, and Applications (DGRC), pp. 172-179, 2018.

[8] D. Silver, J. Schrittwieser, et al., "Mastering the game of go without human knowledge", Nature, vol.550, pp. 354-359, 2017.

[9] T. Iwatani, et al., "Interview about the Origin of Game AI "PAC-MAN"", Journal of the Japanese Society for Artificial Intelligence, vol.34, No.1, pp. 86-99, 2019 (in Japanese).

[10] I. Kokolo and V. Simon, "Production of Various Strategies and Position Control for Monte-Carlo Go - Entertaining Human Players," Information Processing Society of Japan Game Programming Workshop, No.6, pp. 47-54, 2012 (in Japanese).

[11] M. Csikszentmihalyi, Flow: The Psychology of Optimal Experience, New York: Harper and Row, 1990.

[12] C.J.C.H. Watkins and P. Dayan, "Technical Note: Qlearning," Machine Learning, vol.8, No.4, pp. 279-292, 1992.

[13] R.S. Sutton, "Learning to predict by the methods of temporal differences," Machine Learning, vol.3, pp. 9-44, 1988.