6P-03

Update Reward Function based on Accumulated data

Kohei Nakazawa[†] Kenichi Nakazato[‡]

Bosch Center of Artificial Intelligence, University of Southampton[†] Bosch Center of Artificial Intelligence[‡]

i. Introduction

Normal reinforcement learning (RL) without an appropriate reward system takes longer for learning. Then, setting midpoint rewards (sub-rewards) in RL is challenging, but can accelerate RL. A way to settle these 2 problems concurrently is introduced in this paper. Trial data is made use of to set extrinsic rewards, allowing decreases in learning cost and time. Changing the reward system can restrict the agent from an exploration of solution spaces [1]. Hence the way to minimise such restrictions is also explained. This new approach on RL is called "Active reward system." Physical experiments were impractical; hence they were simulated with Python programming. As a RL algorithm, Q-learning was used for this research.

ii. Problem setting

The one-dimensional RL problem "Curling" is set to validate the Active reward system. The game's objective is to stop the stone at the designated range where $x = 6.0 \sim 8.0$. Initial conditions for launching a stone are for $x = 0.0$ at $velocity = 5.0$ (Fig. 1). At every timestep, the agent selects a backwards force from differing options of magnitude to exert on the stone to reduce its velocity. The agent's action is updated by Q learning (Eq.1).

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(s_t, a) - Q(s_t, a_t)\right] \quad (1)$$

The original reward is set as shown in Fig 2. The states (x, v) are discretised and each cell is given an individual reward. This is called a reward table. The rewards -1.0 or +1.0 on the 2nd column and the lowest row are the original rewards and are determined by just the game's rule and not from any prior knowledge or experiments.

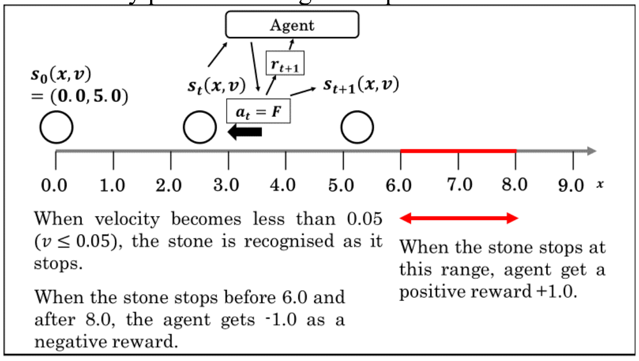
Agent
r_{t+1}
$s_0(x,v)$
$= (0.0, 5.0)$
$s_t(x,v)$
$a_t = F$
$s_{t+1}(x,v)$
0.0
1.0
2.0
3.0
4.0
5.0
6.0
7.0
8.0
9.0
x
When velocity becomes less than 0.05 ($v \leq 0.05$), the stone is recognised as it stops.
When the stone stops before 6.0 and after 8.0, the agent gets -1.0 as a negative reward.
When the stone stops at this range, agent get a positive reward +1.0.

Figure 1 Schematic of "Curling"

x\v	0.0-0.05	0.05-1.0	1.0-2.0	2.0-3.0	3.0-4.0	4.0-5.0
0.0-1.0	-1.0	0.0	0.0	0.0	0.0	0.0
1.0-2.0	-1.0	0.0	0.0	0.0	0.0	0.0
2.0-3.0	-1.0	0.0	0.0	0.0	0.0	0.0
3.0-4.0	-1.0	0.0	0.0	0.0	0.0	0.0
4.0-5.0	-1.0	0.0	0.0	0.0	0.0	0.0
5.0-6.0	-1.0	0.0	0.0	0.0	0.0	0.0
6.0-7.0	1.0	0.0	0.0	0.0	0.0	0.0
7.0-8.0	1.0	0.0	0.0	0.0	0.0	0.0
8.0-9.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0

1.00
0.80
0.60
0.40
0.20
0.00
-0.20
-0.40
-0.60
-0.80
-1.00

Figure 2 Original reward table

iii. Active reward system procedure and theory

The Active reward system is how to produce and update the sub-reward in cells which are originally set to 0.0 in Fig. 2. The process of Q learning with implementation of Active reward system is separated into 3 phases over numerous trials, as illustrated in Fig. 3.

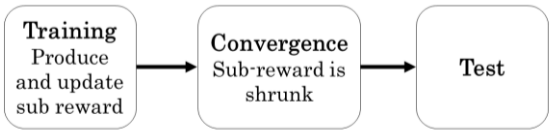

Figure 3 The Procedure of the RL using active reward system

In addition to the original reward, sub-reward is implemented. Though the original reward is kept constant, the sub-reward is constantly updated as the trial progresses.

How to update sub-reward

All trials' trajectories are stored and the expected value for each state on the trajectory is calculated based on how much reward was obtained for each trial. The expected values become sub-rewards $A(s)$ for individual states, and the relationship is given as follows.

$$A(s) = E\left[\sum R(s_t)\right] \quad (2)$$

How to update discounted sub-rewards

Implementing a discount rate (γ_E) for each discretised state decreases the sub-reward of states further from the state which obtains the final reward (Eq.3) (Fig.4). It is updated in similar fashion to the state value function. Eq. 3 shows how the discounted sub-reward is calculated.

$$A_\gamma(s) = E\left[\sum_{t=1}^{t=e} {\gamma_E}^{e-t} R(s_t)\right] \quad (3)$$

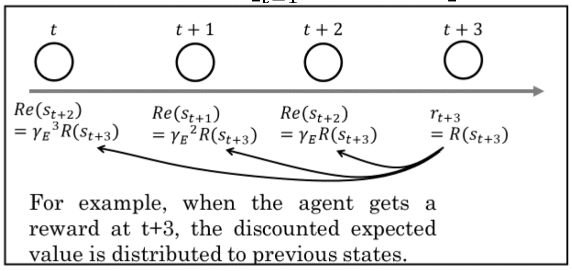
t
$t+1$
$t+2$
$t+3$
$Re(s_{t+2}) = {\gamma_E}^3 R(s_{t+3})$
$Re(s_{t+1}) = {\gamma_E}^2 R(s_{t+3})$
$Re(s_{t+2}) = \gamma_E R(s_{t+3})$
$r_{t+3} = R(s_{t+3})$
For example, when the agent gets a reward at t+3, the discounted expected value is distributed to previous states.

Figure 4 Schematic how A_γ is updated

After the training phase, the sum of the original reward table and sub-reward table(A) gives the total reward table (R). The sub-reward accelerates Q learning and can narrow the exploration space and change the Q table's characteristics compared to Q learning with only the original reward. To minimise these repercussions, the sub-reward is shrunk by reducing hyperparameter $\tau(1 \rightarrow 0)$ which happens in the convergence phase (Eq.4). In this paper, τ decreases linearly with respect to the number of trials in the convergence phase. The number of distributed trials in the convergence phase affects the final test phase's accuracy. For instance, if the number of trials in the convergence phase is too small, then τ decreases sharply. As such, the reward table changes abruptly, and Q-learning cannot adjust in response.

$$R_{t+1} = R_t + \tau A_t \tag{4}$$

An analogy for the sub-reward table is like training wheels for a bike that accelerates a child's learning (Q learning). However, when the child (agent) keeps using them continuously, he or she cannot learn the appropriate way to ride a bike. Therefore, decreasing the sub-reward transfers to the original Q learning can allow the reward system to respond to any situation. Even in the convergence and test phases, the trial data is constantly being used to update the sub-reward.

iv. Result

The simulation of Active reward system was performed using the Eq. 2 on this paper. From results of the success rate and the total reward table ran by actual code implementing "Active reward system", some characteristics of this Q learning were identified. In between 0 to 500 trials, training was performed followed by 150 trials' convergence phase. The reason why the system requires approximately 150 trials for convergence is because as the reward table changes, the Q table needs to adjust accordingly. The normal RL data was performed for comparison. Though normal RL showed a gradual increase of the success rate, Active reward RL gives better improvement and accuracy, as shown in Figure 5.

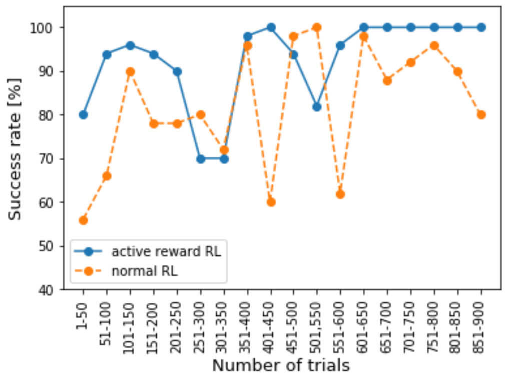

Figure 5 Average success rate for every 50 trials

Fig. 6 shows an updated reward table after 500 trials. The difference between Figs. 2 and 6 are the updated sub-rewards. Then, the final converged graph reward table is shown in Fig. 7, which has a tiny sub-reward in comparison and is almost identical to the original reward table (Fig. 2).

x\v	0.0-0.05	0.05-1.0	1.0-2.0	2.0-3.0	3.0-4.0	4.0-5.0
0.0-1.0	-1.0	0.000	0.000	0.000	0.000	0.455
1.0-2.0	-1.0	0.000	0.000	0.000	0.478	0.372
2.0-3.0	-1.0	0.000	0.000	0.000	0.451	0.090
3.0-4.0	-1.0	0.000	-0.299	-0.149	0.668	0.381
4.0-5.0	-1.0	-0.584	-0.181	0.595	0.991	-1.000
5.0-6.0	-1.0	-0.435	0.659	0.987	0.450	-1.000
6.0-7.0	1.0	1.000	1.000	0.868	-0.911	-1.000
7.0-8.0	1.0	0.986	0.660	-0.953	-1.000	-1.000
8.0-9.0	-1.00	-1.00	-1.00	-1.00	-1.00	-1.00

Figure 6 Total reward table after 500 trials.

x\v	0.0-0.05	0.05-1.0	1.0-2.0	2.0-3.0	3.0-4.0	4.0-5.0
0.0-1.0	-1.0	0.000	0.000	0.000	0.000	0.045
1.0-2.0	-1.0	0.000	0.000	0.000	0.019	0.044
2.0-3.0	-1.0	0.000	0.000	0.000	0.039	0.044
3.0-4.0	-1.0	0.000	-0.014	-0.011	0.045	0.010
4.0-5.0	-1.0	-0.026	-0.012	0.039	0.045	-0.046
5.0-6.0	-1.0	-0.021	0.040	0.045	0.021	-0.046
6.0-7.0	1.0	0.046	0.046	0.039	-0.042	-0.046
7.0-8.0	1.0	0.045	0.030	-0.044	-0.046	-0.046
8.0-9.0	-1.0	-1.0	-1.0	-1.0	-1.0	-1.0

Figure 7 Total reward table at 650 trials (1-500 training, 501-650 convergence)

v. Discussion

This "Active reward system" Q learning is related to data driven offline RL which uses reward sketches [2]. It asks human experts to provide reward models for subsets of the huge offline dataset. From the professional reward model, appropriate reward models are made and applied to all datasets and the training progresses. As it requires human involvement, this might restrict other solutions, or it is difficult to apply this approach to problems unsolvable for humans. In contrast, our approach requires only the original reward setting, which is the final aim of the game or problem. Moreover, our approach does not discourage other solutions since the sub-reward is used just for acceleration of RL and not as a permanent reward.

vi. Conclusion

This paper proposes the new "Active reward system" to be updated by trials' data for RL. This approach was validated on a simple one-dimensional problem. As it can enhance RL with less restrictions for the exploration space, it can contribute to solve a problem when resources are finite. The limitation of this system is that in addition to discount rate and learning rate for Q-learning, it has more hyperparameters such as τ. As these hyperparameters alter the characteristics of the system, determining appropriate hyperparameters for each specific problem will be an extra cost. Exactly how these hyperparameters can affect Q-learning can be ground for future work. The prospect of more effective methods to find sub-rewards or improvements to advance RL based on this research is certainly possible.

vii. Bibliography

[1] Bonsai, "Deep Reinforcement Learning Models: Tips & Tricks for Writing Reward Functions," Bonsai, 16 November 2017.

[2] Serkan Cabi, Sergio Gómez Colmenarejo, et al., "Scaling data-driven robotics with reward sketching and batch reinforcement learning," Robotics: Science and Systems Conference, 2020.

Updated reward function based on accumulated data
† Kohei Nakazawa: Bosch Center of Artificial Intelligence and University of Southampton
‡ Kenichi Nakazato: Bosch Center of Artificial Intelligence