Solve the inverted pendulum problem base on DQN algorithm

Xiaoqian Li[1], Houde Liu, Xueqian Wang

1. department of automation, Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055
E-mail: thuqianqian@gmail.com

Abstract: Inverted pendulum is a classical control problem.Traditional control methods such as PID and fuzzy control can achieve better results.With the rise of artificial intelligence technology, deep learning and reinforcement learning have attracted much attention.Among them, the combination of the two has considerable potential for in-depth reinforcement learning.Q-learning was proposed in 1989 as a classical algorithm in reinforcement learning. In recent years, some scholars proposed the algorithm of Deep learning plus q-learning, Deep Q network(DQN), to solve the problem that q-learning is inherently unable to solve the problem of high-dimensional input, and achieved good results.In this paper, based on VREP simulation environment, DQN is used to try to solve inverted pendulum problem.

Key Words: Reinforcement Learning, DQN,Deep learning, Inverted pendulum

1. INTRODUCTION

Inverted pendulum system is a typical nonlinear system with high order, multivariable, severe instability and strong coupling.First-order inverted pendulum has the advantages of simple structure and easy to simulate. At the same time, it can effectively reflect many key problems in the control process, such as the nonlinearity and robustness of the system, etc., so the research on inverted pendulum has always been an enduring subject in the field of control [1].At present, the control methods of inverted pendulum can be divided into three categories: linear control, predictive control and intelligent control, among which the linear control method is the most widely used [2][3].The linear control method is based on the linear model of the system. For the first and second order inverted pendulum system, due to its small linearization error and simple model, the stability control problem of conventional inverted pendulum can be solved.However, conventional linear control methods are difficult to obtain good results when dealing with nonlinear models.Predictive control method has better control effect in theory, but because of its complexity, high cost and difficulty in real-time application in rapidly changing systems, it is mainly used in simulation experiments at present.However, intelligent control method is a hot research topic at present. It has many kinds of methods, advantages and better control effect for nonlinear controlled objects.
In this paper, artificial intelligence control algorithm(DQN) is used to control the first-order inverted pendulum system.

This work was supported by the National Natural Science Foundation of China (No.61673239, No.61803221), the Science and Technology Planning Project of Guangdong Province (No. 2017B010116001) and the Basic Research Program of Shenzhen (JCYJ20160301100921349, JCYJ20160428182227081).

Firstly, the MDP model of reinforcement learning is briefly introduced, and then the working principle of DQN is expounded. Finally, modeling and simulation are conducted to draw a conclusion

2. MDP and Q-learning

The typical model of reinforcement learning is shown in the Fig 1[4]:

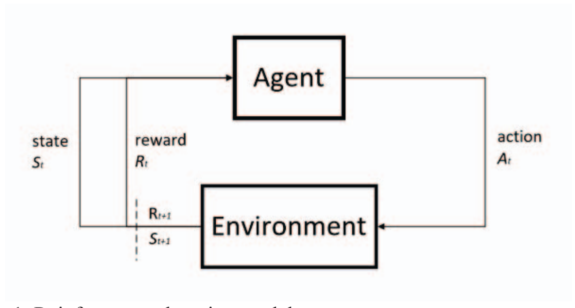

Fig 1. Reinforcement learning model.

The Agent is active in the environment and receives a reward and the next state for the next time step.
MDP is short for Markov Decision Processes. MDP illustrates the assumption that the next state is related only to the current state and not to earlier states. Based on this, we can write the dynamics of the MDP.With the dynamics of the MDP, We can calculate the next state and reward based on the current state and action[4].

$$p(s',r\mid s,a)\doteq P\{S_t=s',R_t=r\mid S_{t-1}=s,A_{t-1}=a\} \tag{1}$$

In reinforcement learning, we hope that the larger the long-term reward an agent can obtain, the better. From the formula, it is[4]

$$G_t\doteq\sum_{k=t+1}^{T}\gamma^{k-t-1}R_k \tag{2}$$

From this, we can calculate the value of a state under strategy pi called a state-value function

$$v_\pi(s) \doteq \mathrm{E}_\pi[G_t \mid S_t = s] \tag{3}$$

It can also calculate the value of an action taken by an agent in a state, which is called action-value function

$$q_\pi(s,a) \doteq \mathrm{E}_\pi[G_t \mid S_t = s \mid A_t = a] \tag{4}$$

If we find the optimal action-value function, then in each state, we can select the action according to the value of action-value function. States and actions are tabulated to get Q-table as shown in Fig 2.

	a1	a2	...	a3
s1	Q(s1,a1)	Q(s1,a2)	...	Q(s1,a3)
s2	Q(s2,a1)	Q(s2,a2)	...	Q(s2,a3)
...	...	...	...	...
sn	Q(sn,a1)	Q(sn,a2)	...	Q(sn,an)

Fig 2. Q-table

We can iterate through the Temporal-Difference to get the Q-table[4]

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha[R_{t+1} + \gamma \max_{a\in A(S_{t+1})} Q(S_{t+1},a) - Q(S_t,A_t)] \tag{5}$$

A method to optimize action-value function by iteratively updating q-table, that is, Q-learning.

There is another method that is very similar to q-learning, called SARSA

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha[R_{t+1} + \gamma Q(S_{t+1},A_{t+1}) - Q(S_t,A_t)] \tag{6}$$

SARSA is very similar to Q-learning. The only difference is that SARSA is an "activist". In the next state, Q-learning is to think, while SARSA is to act first

3. DQN algorithm

As seen above, each (s,a) pair can be found in the Q-table, but such a Q-table is only applicable when the dimensions of s and a are not very large, otherwise, the program is time-consuming to index at run time.

In fact, most of the problems we face are this high state and state dimension.

For example, if the algorithm learns to play Atari games based on image input, the pixels of an image can be tens of thousands of times, and the dimensions of state are too large to count due to the difference of color components.

If we continue to set up Q-table, this method is not operable.The problem can be solved this way.

In fact, the essence of Q-table is a binary discrete function

$$Q(S,A) = f(s,a) \tag{7}$$

Therefore, we can naturally think of a function value approximation method to solve the problem of high-dimensional input.Such as

$$Q(S,A) \approx w_1 s + w_2 a + b \tag{8}$$

If w is used to uniformly mark the parameters of the function, then

$$Q(S,A) \approx f(s,a,w) \tag{9}$$

The function f represented by a neural network as shown in Fig 3.

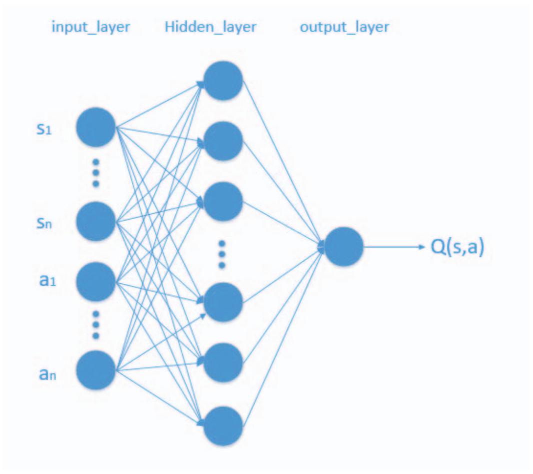

Fig 3. Q-table was approximated by artificial neural network

We can represent the function f by a neural network.

Let's call this network that evaluates q Q_eval_net.

With this network, our next step is to optimize the network, so what is the label of the network output?

In DQN, this network tag is the same thing that you get from bellman's equation, which is also called target_q.

We calculated target_q with a network Q_target_net of the same structure.

The Q_eval_net parameter is then assigned to Q_target_net at every C steps.

This method is called "Fixed Q target."

After solving the problem of poor expansibility of q table, DQN also faces a common problem of reinforcement learning.

Because training data are the agent from the environment interaction, there is no full sample, so the agent is very easy to fall into local optimum, in order to solve this problem, we have the agent interaction are stored by the process of each training network with random or deliberately learning from past experience, so to a certain extent, can solve the problem of local optimum.

In fact, Fixed Q target and experience replay are two important means for DQN to be successful.

To sum up, workflow of DQN algorithm is shown in the Fig4.

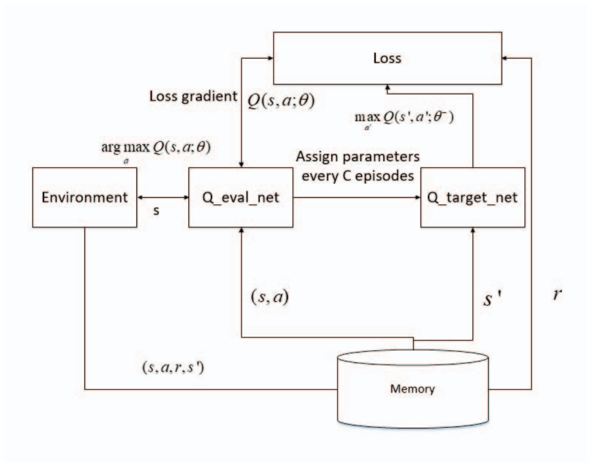

Fig 4. DQN workflow

4. Simulation environment and settings

We consider two environments to test DQN algorithm, OpenAI gym and vrep.

OpenAI gym is an open source reinforcement learning algorithm testing platform that integrates many reinforcement learning environments, among which CartPole environment is what we need.OpenAI gym is very convenient to use. It can directly call the API to get the environment and reward information. However, due to the high degree of integration, it is difficult to customize the environment we want and adjust the input and output parameters.For example, the state output from the environment can be used immediately, regardless of its meaning.But as you can actually tell from looking at the source code, CartPole's state is a few values picked up by kinetic analysis, which means that under normal usage, we lose control of the state setting.

Based on the characteristics of OpenAI gym, it serves as the baseline of our algorithm with stable and unified characteristics.

The Open AI Gym environment is shown in Fig5.

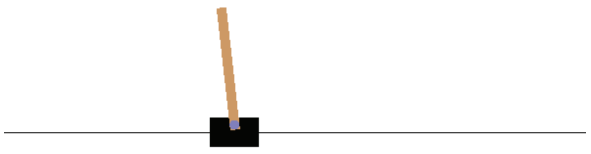

Fig 5. Open AI Gym

VREP (virtual robot experimentation platform)

) is an open source robot simulation platform, it has very high flexibility, almost can customize any information you want, such as the quality of the bar, the friction of the shaft, acceleration, different physical engines, based on VREP reinforcement learning simulation environment can be customized our training parameters.

While modeling in VREP, The simplest scenario is to simply connect a stick with a joint and move horizontally along the joint normal.

However, in the actual test, this way of moving joint directly would be unstable.

So stick is put on the car and the car moves along the x axis.

In VREP, we set different parameters, such as physical engine, VREP has 4 engines, 3 of which are open source, 1 is commercial.

The cart-pole model is shown in the Fig 6.

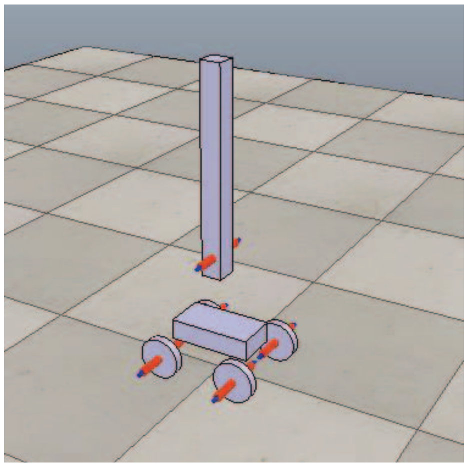

Fig 6. Model in VREP

In order to eliminate interference factors, several points should be noted in modeling

1. The inverted pendulum cannot be set too high, and the base 4 joints speed cannot be set too high. Otherwise, the base may slip at the moment of movement.
2. The masses of base and wheels can be larger and those of inverted pendulum are smaller within a reasonable range, which can eliminate the phenomenon of base deviation caused by inertia.
3. The physics engine should be Vortex, because Vortex is the most authentic and stable engine in VREP.
4. Because python client and VREP server have latency (usually in dozens of ms), try to use the synchronous mode of VREP, otherwise there will be large interference in asynchronous mode, and there will be some methods to lower the gravity and so on to eliminate the interference.

Here are some parameters to set

Table1. Simulation setting

Base mass	100kg
Wheel mass	5kg
Motor torque	100Nm
Stick length	0.5m
Stick mass	1kg

Physics engine1	Vortex
Physics engine2	bullet
Physics engine4	ODE
Physics engine4	Newton
Simulation step time	50ms

5. Experimental Result

First let's look at the results in the Open AI Gym.
In the training of Open AI Gym environment, we can set the learning rate, the frequency of changing target net parameters, the degree of greedy strategy, the size of memory bank and so on.These parameters
Experiment 1:

Table2. Training parameters

Learning rate	0.01
Epsilon greedy	0.9
Replace target net parameters	100/episodes
Memory size	2000

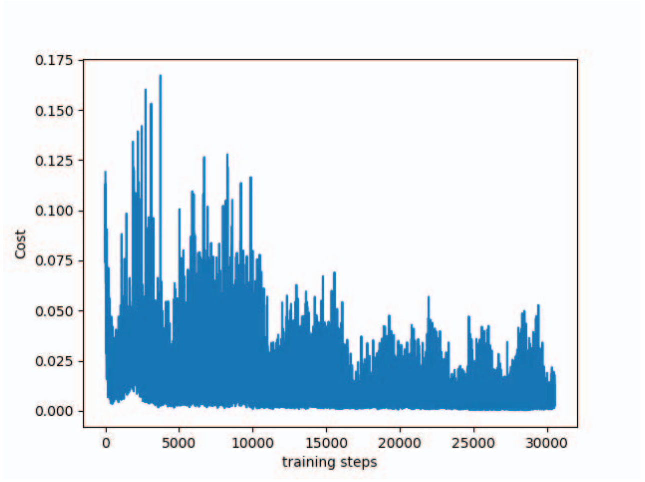

Fig 7. Experiment 1 result
Experiment 2:

Table3. Training parameters

Learning rate	0.01
Epsilon greedy	0.5
Replace target net parameters	100/episodes
Memory size	2000

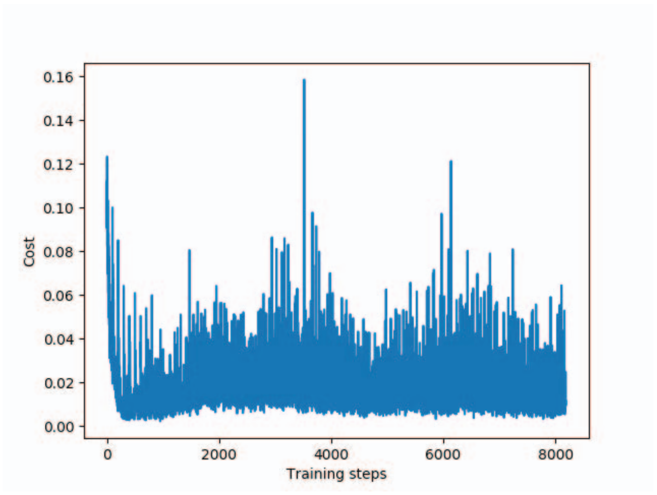

Fig 8. Experiment 2 result
Experiment 3:

Table4. Training parameters

Learning rate	0.01
Epsilon greedy	0.9
Replace target net parameters	10/episodes
Memory size	2000

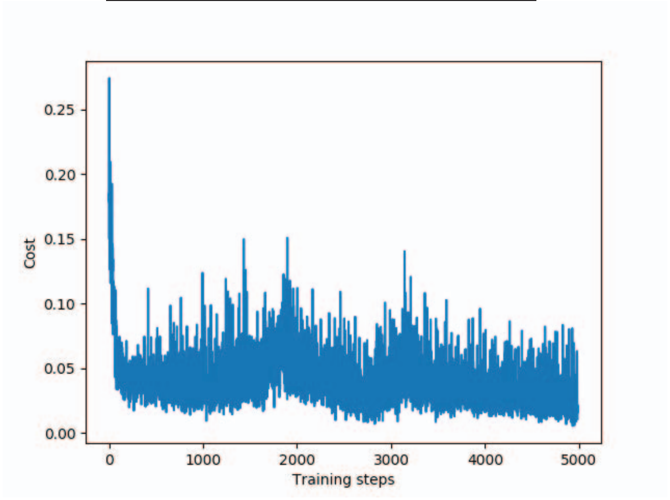

Fig 9. Experiment 3 result
Experiment 4:

Table5. Training parameters

Learning rate	0.01
Epsilon greedy	0.9
Replace target net parameters	100/episodes
Memory size	200

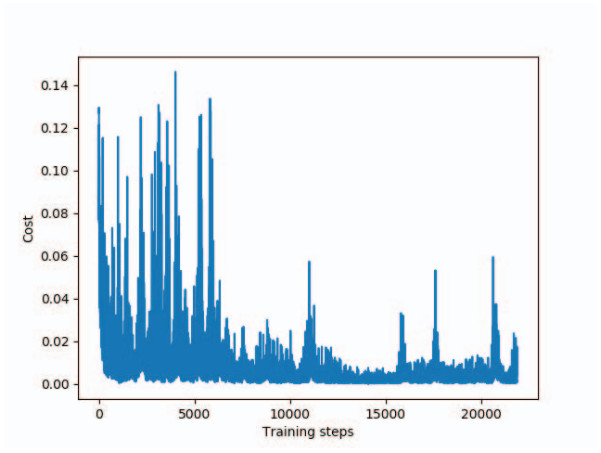

Fig 10. Experiment 4 result

As for setting up in VREP, there are many issues to consider . This time we will use the reward value to measure the results of the algorithm.
In the experiment, stick's deflection Angle was first selected as state, and only this one was selected.
The reason why I chose this is because, one, intuitively this quantity of state is a good indicator;
Second, use the least resources to achieve better results.
Base of the four motors rotation direction as action.
These two parameters are selected to be sent to DQN network for training.
Hyper parameters are shown in Table6.

Table6. Program parameter settings

episode	1000
Batch size	32
optimizer	AdamOptimizer
Learning rate	0.0001
Test interval	50
Reward	1 if not done
Done flag	Pendulum orientation>10°
Base motor velocity	5rad/s
State dimensionality	1

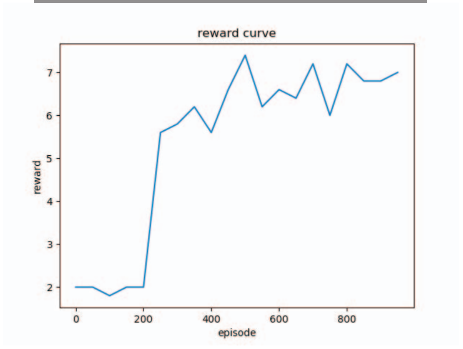

Fig 11. Reward curve
It can be seen from the figure that only the attitude of inverted pendulum is selected as the input of state, and the rotation direction of base motor is used as the input of action, the results can converge to a good level in 1000episode.

If we use more state parameters to feed the network, such as increasing the speed of stick and tick, we will increase the speed of rotation.
So we get this program parameter settings. As shown in Table7.

Table7. Program parameter settings

episode	10000
Batch size	128
optimizer	AdamOptimizer
Learning rate	0.0001
Test interval	50
Reward	1 if not done
Done flag	Pendulum orientation>10°
Base motor velocity	5rad/s
State dimensionality	3

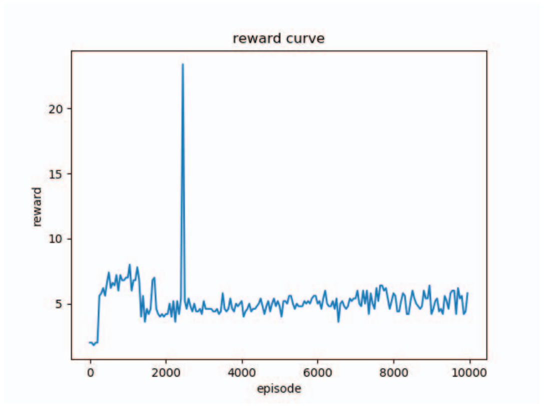

Fig 12. Reward curve
What if we change the reward setting?
Reward is one of the main driving forces of reinforcement learning algorithm. This time, we change the reward to be the reciprocal of stick deflection Angle squared, and the input state is the speed of stick, the Angle of stick, and the rotation speed of stick.

Table8. Program parameter settings

episode	200
Batch size	16
optimizer	AdamOptimizer
Learning rate	0.0001
Test interval	1
Reward	1/ square (stick orientation)
Done flag	Pendulum orientation>10°
Base motor velocity	5rad/s
State dimensionality	1

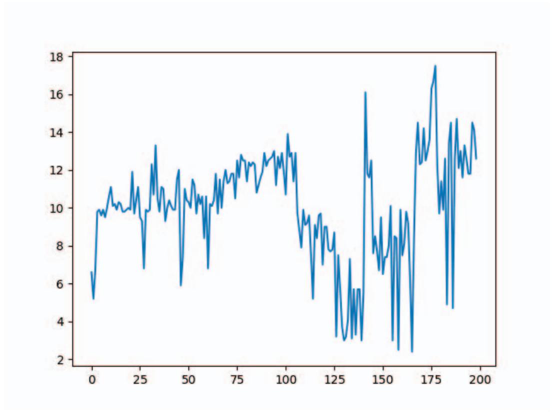

Fig 13. Reward curve

6. Conclusion

DQN algorithm runs successfully on both simulation platforms. However, since VREP USES real physics engine, there are many variables to be considered and many perturbations occur, so the overall effect of VREP is inferior to that of Open AI Gym.But they all work.The Open AI Gym four experiments, we use the control variable method to change the four parameters, from the perspective of the cost curve of output, the memory size and Episilon greedy two parameters are significantly affected the cost of the final value, it is because these two parameters optimization, can let the algorithm in the training experience more case, rather than the case near the local optimum.As the frequency of replacing Q_target_net parameter increases, the decreasing speed of cost increases, which is also easy to understand. The network parameter is updated more frequently, and the convergence speed increases.

REFERENCES

[1] Yingjun Sang, Yuanyuan Fan,Caiqian Xu,. Study on the control method of single-stage inverted pendulum[J].Control Engineering,2010(6):743-750

[2] Lingfang Sun,Hui Kong,Changguo Liu, Inverted pendulum system and its research status[J]. Machine tools and hydraulics,2008(7):306-310

[3] Lingfang Sun,Jijun Wang, Overview of inverted pendulum system[J]. Control theory and application,2011(2):1-5

[4] Richard S.Sutton and Andrew G.Barto. Reinforcement Lear ning :An Introduction[EB/OL]. http://incompleteideas.net/book/the-book-2nd.html ,2018-1 1-10

[5] Gang Yao, There are 3 states, which are the speed of stick and the speed of stick,2012(24): 31

[6] Baocai Yin,Wentong Wang,Lichun Wang, Review of deep learning research[J]. Journal of Beijing university of technology,2015(1):48-59

[7] Guo Lili,Ding Shifei.Progress in deep learning[J].computer science,2015(5):28-33

[8] Zhang Rubo,Gu Guochang,Liu Zhaode,Wang Xingce.Reinforcement Learning Theory, Algorithm and Application[J].Control Theory and Applications,2000(5):637-642