LEARNING POLICY GRADIENT: A METHOD FOR REINFORCEMENT LEARNING

Mincong Luo
Information Centre
China Institute of Atomic Energy
Peking,China
luomincentos@ciae.ac.cn

ABSTRACT

One less addressed issue of deep reinforcement learning is the lack of generalization capability based on new state and new target, for complex tasks, it is necessary to evaluate the correct strategy based on the current state. In this paper we present an approach that can let agent to learn the policy gradient based on the current state and the actions set, by which the agent can learn a policy map with generalization capability. we evaluate the proposed method on the 3D enviroment Fuzzy World compared with the baseline model, detailed experimental results and proofs are also given.

1 INTRODUCTION TO DRL

In recent years the deep reinforcement learning(DRL) methods haveshown promising results in virtual enviroment tasks and real-world tasks Sutton & Barto (1992) Mnih et al. (2016) Babaeizadeh et al. (2016) Tai et al. (2017). The study of reinforcement learning is based on the scope of classical physics. The time of the world can be divided into time slices, and there is a complete sequence, which is the series of time series states mentioned above, one of the important assumptions is that each time the parameter adjustment has a deterministic effect on the environment, ie the input is determined and the output is determined Hussein et al. (2018) Zhang et al. (2018) Lu et al. (2016).

$$\{s_1, a_1, r_1, s_2, a_2, r_2 ...\}$$

The Bellman equation Bao et al. (2008) illustrates the relationship between the value function of the current state and the value function of the next state. From a formula perspective, the value of the current state is related to the value of the next step and the current feedback, Reward Baxter et al. (2011) Wang et al. (2016) Rohrbach et al. (2015). It shows that the Value Function can be calculated by iteration, which is the basis for intensive learning:

$$v(s) = \mathbb{E}[R_{t+1} + \lambda v(S_{t+1})|S_t = s]$$

The Action-value function $Q^\pi(s, a)$ is defined to denote the cumulative reward of strategy π starting from state s, after performing action a:

$$Q^\pi(s, a) = \mathbb{E}[r_{t+1} + \lambda r_{t+2} + \lambda^2 r_{t+3} + ...|s, a] \tag{1}$$

$$\tag{2}$$

$$= \mathbb{E}_{s'}[r + \lambda Q^\pi(s', a')|s, a] \tag{3}$$

Q-Learning is based on value iteration. There is virtually no way to traverse all states, and all the actions, but only a limited series of samples. Therefore, only limited samples can be used for operation. A way to update the Q value is proposed in Q-Learning:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha(R_{t+1} + \lambda \max_a Q(S_{t+1}, a) - Q(S_t, A_t))$$

We define $\rho(\pi)$ as the average reward formulation that is useful in the next, in which policies are ranked according to their long-term expected reward per step:

$$\rho(\pi) = \lim_{n\to\infty} \frac{1}{n}\{r_1 + r_2 + ... + r_n|\pi\} = \sum_a d^\pi(s) \sum_a \pi(s,a) R_s^a$$

where the $d^\pi(s) = \lim_{t\to\infty} Pr(s_t = s|s_0, \pi)$ is the stationary distribution of the all states under policy π.

2 THE THEOREM OF POLICY GRADIENT

The policy gradient $\frac{\partial\pi(s,a)}{\partial\theta}$ is what we wanna the agent to learn because the agent can update the policy based on the new states and targets by which. But first of all establishing the relationship between the policy gradient $\frac{\partial\pi(s,a)}{\partial\theta}$ and the globle value is needed. To address this issue the following theorem is proposed;

Theorem 1 *In the MDP(Markov Decision Process) for average-reward formulations:*

$$\frac{\partial\rho}{\partial\theta} = \sum_a d^\pi(s) \sum_a \frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a)$$

Proof 2.1 *For the average-reward formulations, we can denote the reward as:*

$$V^\pi(s) = \sum_a \pi(s,a) Q^\pi(s,a)$$

so we can get:

$$\begin{aligned}
\frac{\partial V^\pi(s)}{\partial\theta} &= \frac{\partial}{\partial\theta} \sum_a \pi(s,a) Q^\pi(s,a) \\
&= \sum_a (\frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a) + \pi(s,a) \frac{\partial Q^\pi(s,a)}{\partial\theta}) \\
= \sum_a (\frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a) &+ \pi(s,a) \frac{\partial}{\partial\theta}(R_a^s - \rho(\pi) + \sum_{s'} P_{ss'}^a V^\pi(s'))) \\
&= \sum_a (\frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a) + \pi(s,a)(-\frac{\partial\rho}{\partial\theta} + \sum_{s'} P_{ss'}^a \frac{\partial V^\pi(s')}{\partial\theta}))
\end{aligned}$$

which can be sorted out and summing by d^π as:

$$\sum_s d^\pi(s) \frac{\partial\rho}{\partial\theta} = \sum_s d^\pi(s) \sum_a \frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a) + \sum_s d^\pi(s) \sum_a \pi(s,a) \sum_{s'} P_{ss'}^a \frac{\partial V^\pi(s')}{\partial\theta}) - \sum_s d^\pi(s) \frac{\partial V^\pi(s)}{\partial\theta}$$

using the stationary of d^π and we can get:

$$\begin{aligned}
\sum_s d^\pi(s) \frac{\partial\rho}{\partial\theta} &= \sum_s d^\pi(s) \sum_a \frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a) + \sum_{s'\in S} d^\pi(s') \frac{\partial V^\pi(s')}{\partial\theta} - \sum_{s\in S} d^\pi(s) \frac{\partial V^\pi(s)}{\partial\theta} \\
&\Rightarrow \frac{\partial\rho}{\partial\theta} = \sum_a d^\pi(s) \sum_a \frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a)
\end{aligned}$$

Based on this theorem, we can use the $\frac{\partial\rho}{\partial\theta} = \sum_a d^\pi(s) \sum_a \frac{\partial\pi(s,a)}{\partial\theta} Q^\pi(s,a)$ to find the most suitable descent gradient. In the next section, we will propose a model for this.

3 THE MODEL AND THE EXPERIMENT

The $\rho(\pi)$ is defined as the average reward formulation denotes the performance of the polocies the agent chooses, which is an optimization target needs to be maximized. So the base idea is:

$$\frac{\partial \pi(s,a)}{\partial \theta}^{*} = \min_{\frac{\partial \pi(s,a)}{\partial \theta}} \rho(\pi)$$

where the $\frac{\partial \pi(s,a)}{\partial \theta}^{*}$ means the gradient maximize the $\rho(\pi)$, so the core idea is let $\frac{\partial \rho}{\partial \theta} = 0$ and this means $\frac{\partial \pi(s,a)}{\partial \theta}$ and $Q^{\pi}(s,a)$ need to be orthogonal:

$$\frac{\partial \rho}{\partial \theta} \to 0 = \sum_{a} d^{\pi}(s) \sum_{a} \frac{\partial \pi(s,a)}{\partial \theta} Q^{\pi}(s,a)$$

The Model: The model with two approximation functions, $f^{\triangledown}(\cdot)$ and $f^{Q}(\cdot)$, is proposed. The $f^{\triangledown}(\cdot)$ tries to learn the map from the states and actions to the gradient, and $f^{Q}(\cdot)$ tries to evaluate the value of the actions based on the states. The whole structure of the model is given in the Figure.1.

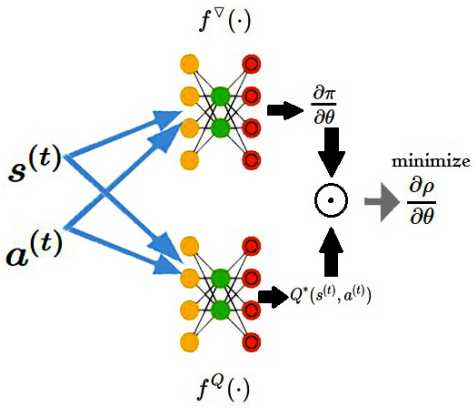


Figure 1: The model is based on the action a and state s with two approximation functions $f^{\triangledown}(\cdot)$ and $f^{Q}(\cdot)$.

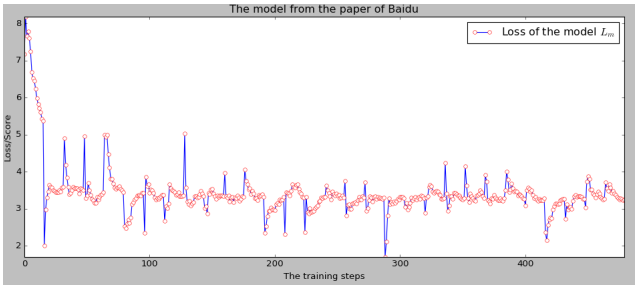
The Dateset: We make the experiment enviroment on the Fuzzy World, in which a 3D virtual enviroment with multiple tasks that require logical reasoning based on visual information and semantic information can be given.

The Baseline Model: The baseline model we use is the GFT(Guided Feature Transformation) model proposed by Baidu Yu et al. (2018),which is a simple but effective neural language grounding module for embodied agents that can be trained end to end from scratch taking raw pixels, unstructured linguistic commands, and sparse rewards as the inputs. And the experiment enviroment is also on the Fuzzy World.

The Result: The result shows that our method (as the target cost function) can better converge more stably and quickly compared to the baseline model.

4 CONCLUSION

In this paper, we propose a method for the gradient learning which can be seen as the automatic policy gradient descent. As experiments established, the approach is powerful and did better than specifically fitted solutions such as the weak-supervised or half-supervised models like GFT(Guided Feature Transformation) model proposed by Baidu Yu et al. (2018). $f^{\triangledown}(\cdot)$ is used to learn the policy gradient, which can be pre-trained as a network and used in the agent tasks. However, doubtlessly much more work can be done on this front by more tasks.



(a) Overview

(b) Representation Vector

Figure 2: **(a) : The loss curve of the baseline**. In the task, the loss drops very quickly but the convergence is very poor and the fluctuation is very large. **(b) : The loss curve of the proposed model**. This method (as the target cost function) for the task can better converge more stably and quickly. Notice the loss function is not the same so the loss value can't be be in the same range of values.

REFERENCES

Mohammad Babaeizadeh, Iuri Frosio, Stephen Tyree, Jason Clemons, and Jan Kautz. Reinforcement learning through asynchronous advantage actor-critic on a gpu. 2016.

Bing Kun Bao, Bao Qun Yin, and Hong Sheng Xi. Infinite-horizon policy-gradient estimation with variable discount factor for markov decision process. In *International Conference on Innovative Computing Information and Control*, pp. 584–584, 2008.

Jonathan Baxter, Peter L Bartlett, and Lex Weaver. Experiments with infinite-horizon, policy-gradient estimation. *J Artificial Intelligence Research*, 15(1):351–381, 2011.

A. Hussein, E. Elyan, M. M. Gaber, and C. Jayne. Deep imitation learning for 3d navigation tasks. *Neural Computing and Applications*, 29(7):389–404, 2018.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical co-attention for visual question answering. 2016.

Volodymyr Mnih, Adri Puigdomnech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. 2016.

Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. pp. 817–834, 2015.

R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction. *Machine Learning*, 8(3-4): 225–227, 1992.

Lei Tai, Giuseppe Paolo, and Ming Liu. Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation. 2017.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. 2016.

Haonan Yu, Xiaochen Lian, Haichao Zhang, and Wei Xu. Guided feature transformation (gft): A neural language grounding module for embodied agents. 2018.

Tianhao Zhang, Zoe Mccarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. 2018.